Bridging the Gap Between Virtual and Physical Classrooms to Enhance Distance Learning Experience

Christos Goumopoulos
Hellenic Open University & Aegean University
Patras & Samos, Greece
goumop@{tutors.eap.gr, aegean.gr}

Nikos Kokkos, Ch. Karachristos, Achilles Kameas
Hellenic Open University
Patras, Greece
nkokkos@eap.gr; karachrist@eap.gr; kameas@eap.gr

***Abstract*—In this paper we report on our current research and development work that aims to apply pervasive computing and context-aware technologies to enhance the teaching and learning experience in distance education environments. In order to achieve this goal we move from the 2-D user interfaces found in virtual class management systems to the ubiquitous 3-D environment of the classroom. A smart classroom prototype is presented which provides the necessary mechanisms for building context-aware services to enrich distance learning experience. The fusion of location tracking with activity recognition mechanisms allows to control the liveliness of the video stream transmitted to remote students in order to reduce the gap between physical and virtual classrooms.**

Keywords—location tracking; smart classroom; distance learning; pervasive computing; context-awareness; filtering;

I. INTRODUCTION

Distance learning refers to an educational methodology that aims at delivering education to students through the employment of modern information and communication technologies instead of regularly attending courses in traditional classrooms [1]. Distance education systems are categorized as asynchronous and synchronous. The former provide delayed interaction whereas the later provide real-time interaction between tutors and students [2].

A smart classroom is a physical classroom in which information and communications technologies have been integrated. The technological infrastructure of a typical smart classroom includes equipment for multimedia content rendering, projectors, large screens for video projection, computers, smart boards and desks, data centers, etc. This infrastructure facilitates students to watch a lecture without concerning on keeping detailed notes, since after the lecture they can have access to the teaching material including notes and annotations.

By applying pervasive computing and context-aware technologies in a smart classroom a tele-education experience similar to a real classroom experience can be provided [3]. For example, location-awareness combined with activity recognition and multimodal interfaces can enhance the services and interactions provided in a smart classroom. In such an intelligent environment students can interact with the material, the instructor and the other students in new ways [4].

In this paper we describe the design and implementation of a context-aware application that controls the precision of the video stream transmitted to remote students when a course is taking place within a smart classroom with an aim to enhance the learning experience of distance students. In particular we examine the following application scenario. A classroom is equipped with cameras which aim to provide a "live" image of the classroom to the remote students. The tutor wears a special card that communicates with a number of position sensors that are mounted in a number of places in the classroom. This infrastructure and the associated software mechanisms allows the accurate location tracking of the tutor and the development of location-aware applications and services. The application developed can give commands in the camera to rotate in 300 degrees and to focus to the tutor when he speaks, depending on her current position. In addition, by exploiting other sources of context within the classroom, more refined control can be provided so that when the tutor writes something on the board the camera can focus on the board, or if the tutor holds a physical object showing it to the students then the camera is focusing to the object. The system has been developed to meet the requirements of the students of Hellenic Open University (HOU), but can have broader use in distance learning settings.

The rest of the paper is organized as follows. Section II presents the smart classroom infrastructure developed and discusses the associated design issues. Section III provides a detailed presentation of the main components developed. First we discuss the location tracking system and the filtering technique developed in order to achieve the required location tracking precision. The necessity of a distribution middleware that integrates the location tracking platform with context-aware applications is discussed next. The camera control logic exploiting location and other context is explained followed by the video streaming solution developed for supporting the smart classroom application prototype. Finally, a preliminary evaluation of our current accomplishments is provided. Section IV presents related work followed by our conclusions in the final section.

II. SMART CLASSROOM INFRASTRUCTURE

Modern virtual learning environments support, among other things, virtual classroom systems for supporting synchronous communication between tutors and learners (e.g. ElluminateLive, Centra Symposium, Macromedia Breeze Lotus LearningSpace, etc.). Most virtual classroom systems are based on the learner-instructor interaction model via a computing device, which means that users need to be static in front of the computer screen and use their keyboard and mouse to participate in the educational process (e.g. presentation of educational content, interaction among participants, and so on). This mode of

interaction is, however, far away from the natural way of interaction in a physical classroom. In a real classroom the tutor can move freely, interact with students in a natural way, using for example gestures, and present his thoughts and ideas by using the whiteboard. This gap between physical and virtual classrooms affects the immediacy and efficiency of both the educational and learning activity.

The design and implementation of a smart classroom, which integrates pervasive computing technologies as an infrastructure for developing context-aware services and applications can lift partially the above restrictions. Such a smart classroom will provide the ability to use natural and multimodal interfaces and context-aware applications supporting the educational process conducted within the smart classroom, as well as interactive e-learning applications supporting with the most natural possible way the educational activity between the local tutor in the smart classroom and the remote students.

HOU has developed a smart classroom prototype which would play a dual role in the context of its activities: (a) researching and fostering opportunities for enhancing teaching and learning in distance learning programs through pervasive computing technologies, and (b) providing a real testbed for developing and testing pervasive computing applications in the context of project assignments. The concept of the smart classroom is shaped upon an ambient intelligent environment which supports three major objectives of the educational process: course creation and presentation, classroom management and student assessment and collaboration. The HOU smart classroom prototype, currently, incorporates equipment such as audiovisual content recording and processing system (server, sound processing and video processing software), digital content distribution / streaming, a sensor package to measure environmental parameters (temperature, light, humidity, noise level etc.), a location tracking system, RFID-based infrastructure, biometric devices, a voice recognition and text rendering system and an eye tracking system for building affect-sensitive intelligent tutoring systems.

In addition to its traditional use as an educational space (e.g. delivering courses), the smart classroom can be used to support other activities such as: videoconference, lecture capture, enhancing the learning experience and supporting research activities. The design and development of the pilot smart classroom required addressing several issues, such as physical infrastructure, security, technology management / support, integration with existing data management systems and deployment of enabling technologies. Fig. 1 illustrates the basic design issues of the smart classroom and summarizes the key elements of each category.

Fig. 2 depicts the smart classroom setup for the needs of the work described in this paper. This is a typical classroom of 66 m2 equipped with smartboards, projectors, cameras, microphones, environmental and location sensors.

III. System Components

A. Location Tracking

Location is one of the major components of the user's context in ubiquitous computing applications [5]. For indoor localization there is a multitude of location tracking technologies with accuracy ranging from centimeters to tens of meters, each of which is appropriate for a particular situation and need [6]. Modern location tracking systems use various techniques such as proximity, triangulation (e.g., lateration, hyperbolic lateration, angulation) and fingerprinting. Algorithms that estimate the location of an entity calculate its distance to a number of reference points by using various characteristics of the received signal such as Received Signal Strength Indicator (RSSI), Time of Arrival (TOA), Time Difference of Arrival (TDOA) and Angle of Arrival (AOA).

Our application scenario requires an average positioning accuracy of about 20 cm for both static and moving objects. In addition, the sensors should be lightweight and portable /wearable (e.g. a card that is wore by a tutor). We have selected to build our location tracking on the Ubisense platform [7]. Ubisense is a commercial product that provides high position detection accuracy (99% within 0.3 m) of objects and people by making use of active tags (*ubitags*), which are small hand-held devices (that can also be attached to the body) and networked sensors distributed in a space. The minimum number of sensors required for the detection of the tag is just two. Ubitags send ultra wide band (UWB) pulses to the fixed sensor receivers inside a local area of interest (cell).

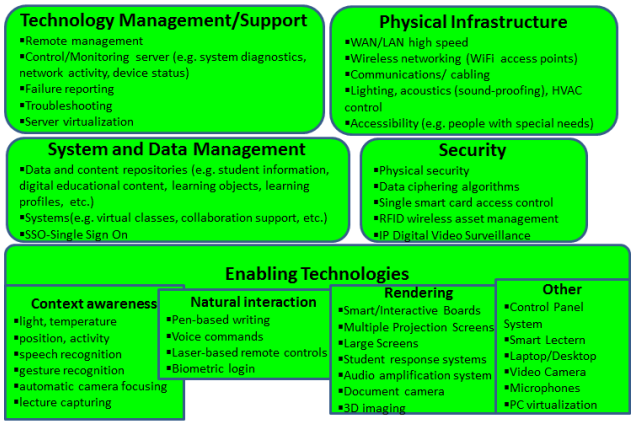

Figure 1. Smart classroom design issues.

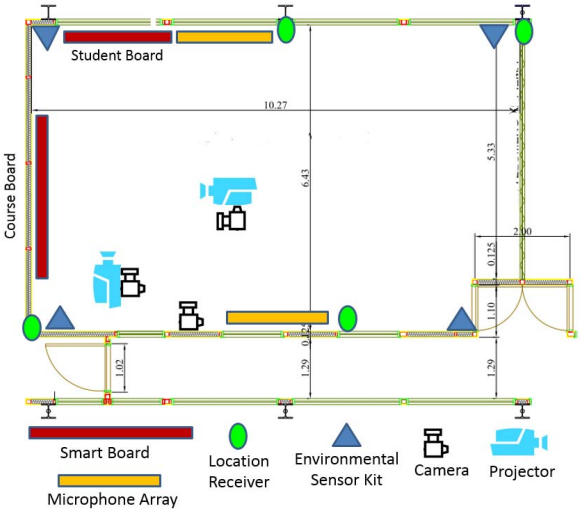

Figure 2. Smart classroom layout.

A combination of TDOA and AOA techniques are used to estimate the position of each ubisense tag (Fig. 3). Improved accuracy of the location tracking system is achieved by applying filtering techniques as discussed in the following section.

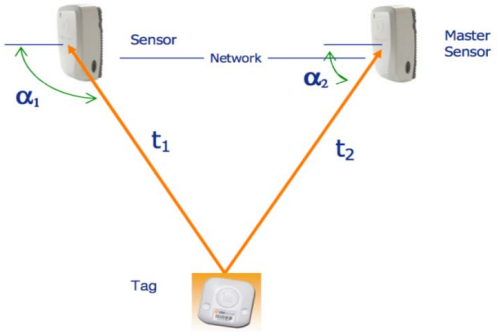

Figure 3. Ubisense sensors tracking ubitag.

Ubitags can be either attached to person's body or objects or carried around to a point of interest. Ubisense provides a proprietary system that consists of a software platform activated by a license scheme. The platform has the characteristic of providing a middleware software solution that can manage real time location information through services as well as simplify the development of location-aware applications through an extensive application programmer interface employing C#/C++ programming libraries. When a tag emits UWB, the signal is picked up by one or more sensors in the cell. The sensors decode the UWB signal and send the angle of arrival and timing information back to the sensor configured as master. The master sensor accumulates all sensed data, computes location of the tag, and delivers the location of the configured sink address. Usually, UWB will penetrate some partition walls such as cinder block or plaster board, but it will not penetrate metallic or water filled bodies or walls. A sensor must see a direct radio signal from a tag to locate the tag, so some consideration is needed of the region to be tracked when planning to install the sensors. Obstructions can include reflecting materials such as metal coated glass panels, partition walls with embedded metal grids or large metallic obstructions.

After the sensors have been installed at certain points indoors, the exact position of the sensors in 3-D coordinates has to be measured and then entered to the configuration console. After that one may choose different methods to calibrate the system ranging from taking an active tag to a number of predetermined positions within the tracked area and running the calibration procedure to running dual calibration where the system instructs you to move a ubitag around for an automatic calibration procedure to take place.

Ubisense provides a number of tools to create location event alerts such as the Site Manager where one can create a virtual space within the preconfigured scan space and also assign easily remembered name to ubitags ids. For example, one may choose to assign the name of Peter to ubitag with id: 010-000-074-034 as opposed to the full id of the ubitag that has to be entered through the location configuration software in order to be activated.

B. *Filtering*

We have applied a filter in an attempt to improve the precision of the location tracking system. A filter is a mathematical instrument that formulates a dynamic model the application of which aims to diminish the noise of sensor measurements. The Kalman and particle filters are two well-known filtering techniques which are used in location tracking systems [5]. In our case we used the particle filter.

We consider as state variables in the case of our location tracking in the 2-D plane to be the positions x_t, y_t and the velocities $u_{x(t)}$, $u_{y(t)}$ at time t. The particle filter represents a vector of potential states, which are considered as particles. Each particle is considered as a hypothesis for the actual state. Its cogency is expressed as a function of the current measurement. After each measurement, the most plausible particles survive and are subject to random change of their state on the basis of a dynamic probabilistic model.

The particle filter is based on a sequence of unknown state vectors x_i and measurements vectors z_i. It uses a probability distribution $p(z_i|x_i)$, which represents the noisy measurements and gives the relationship between the actual state x_i with the measurement z_i. Furthermore, the filter uses the state probability distribution $p(x_i|x_{i-1})$, which gives the distribution of the current state x_i based on the previous state x_{i-1}, and thus represents the dynamics of the system. Finally, the filter uses the initial distribution of the state vector $p(x_0)$, before applying any dynamics.

The key implementation steps of the particle filter, following the implementation of Gordon [8], are the following:

i) *Filter initialization*: $p(x_0) = N(z_0, R_i)$ where R_i is the covariance vector that we introduce for the filter.

ii) *Calculation of particle weights*: $\tilde{w}_i^{(j)} = p\left(z_i|\tilde{x}_i^{(j)}\right)$

iii) *Filter estimation*: $\hat{x}_i = \sum_{j=1}^{N} \tilde{w}_i^{(j)} \tilde{x}_i^{(j)}$

iv) *Selection (particle sampling on the basis of their weight):* This step is sampling of N particles from existing based on the weight of each particle. The selection of the new particles is random where the probability to select a particle is proportional to its weight. This step tends to exclude "weak" particles, while not precluding the option of same value particle more times if the weight is relatively large. This step gives us the particles that will be used in importance sampling step. In our algorithm the "roulette wheel selection" [9] was used due to its simplicity and popularity.

v) *Importance sampling*: This step generates a random set of particles subject to the distribution $p(x_i|x_{i-1})$.

In our implementation we initialize all the particles to have equal weight and the same initial state:

$$p(x_0) = \{X^k, w^k\}_{k=1}^{K} = \left\{ \begin{bmatrix} x_0 \\ 0 \\ y_0 \\ 0 \end{bmatrix}, \frac{1}{K} \right\}$$

where x_0 and y_0 are tag's original, accurate location along the x and y axes with zero initial velocity, and K is the number of samples in our case K = 1000. Here the noise model is chosen to be zero-mean Gaussian random variable with standard deviation σ_{dx} , σ_{dy} along the x and y axis.

Since we assume that the user is moving with a constant velocity we can associate each measurement of Ubisense system with the real location. The velocity is calculated by dividing the distance for each section covered by the time that the user with the tag is on the move. Therefore, multiplying the velocity with the time of measurement, we can calculate the actual location of the user at that moment. This is modelled by the following equations:

$$x_t = \frac{D_x}{T} \times t \qquad y_t = \frac{D_y}{T} \times t$$

where x_t, y_t are the actual positions at time t, D_x, D_y are the actual distances along the x and y axes, and T is the total time for the completion of the recording of a segment of the path.

Then after recording the measurements of the Ubisense location system we apply the particle filter. To assess the performance of the filter we compute the average Euclidean distance between the filter's data and the actual data for the set of metrics. This distance is known as position error (PE) and is defined as follows:

$$PE = \frac{1}{N} \sum_{i=1}^{N} \sqrt{(x_i - \tilde{x}_i)^2 + (y_i - \tilde{y}_i)^2}$$

where x_i, y_i are the real measurements and $\tilde{x}_i$, $\tilde{y}_i$ the values that gives the filter for the i^{th} measurement, and N is the total number of measurements.

Fig. 4 depicts the Euclidean distance between real and filtered by particle filter for each measuring point.

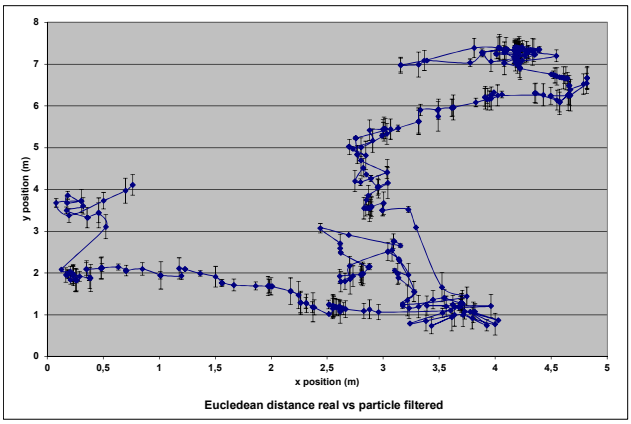

Figure 4. Euclidean distance between real and filtered by particle filter for each measuring point.

The average accuracy of the raw measurements is approximately 26 cm. Experiments with a range of σ_d (the dynamic noise in the movement model) of 1 to 50 cm/s (best results were taken for σ_d = 7 cm/s) revealed that the particle filter improved the accuracy of the raw measurements by approx. 4 cm on average (approximately 15% improvement over raw measurements).

C. Distribution Middleware

There is a need to integrate location data coming from ubitags to external partner clients that need to process that information in order to drive other processes. Ubisense provides access to ubitag coordinate system through an API that presents a minimal sensor abstraction. For example, using minimal programming commands in C#, a programmer may request to subscribe to transaction events related to ubitag movement. The events received provide ubitag identification and coordinates in a 3-D coordinate system assembled in an easily parsed format. Ubisense is built around a Transactional Event Substrate. State such as the locations of ubitags, is cached locally by Ubisense client applications, such as those using Ubisense Client API. Changes to state are multicast to those clients that cache the state. A single transaction can contain a number of individual changes but the transaction is applied atomically to the client side cache of the state and the client application will never see a partially applied transaction.

Ubisense framework provides access to ubitags' coordinate information through dlls. Those dlls must be included into a windows executable or into a .net web service. Our proposed solution is to employ a .net web service that provides ubitags' transactional events to interested parties through a distribution middleware. The main purpose of the middleware is to provide scaling capabilities to the interested subscribed clients, sensor abstraction, and data transmission using open standard format such as JSON.

The distribution system is built upon RabbitMQ[1], an industry proven open source messaging middleware. RabbitMQ has a lot of different libraries for multiple operating systems and gives the option to the programmer to create multiple message queues depending on the required data streams. For our case, we chose to create JSON messages based on Ubisense transactional responses and drive those messages through data exchange points assigned by RabbitMQ. Creating exchange points in RabbitMQ gives the ability to interested parties/clients to subscribe to those points and receive the data. JSON message creation was implemented on the Ubisense client application which was built by using Ubisense .dll library callbacks. For our testing purposes we created a message exchange of type "fanout" called ubisense-data. By giving to our exchange type the characteristic of "fanout", the exchange point gains the capability to subscribe its data to multiple clients. Of course, for multiple ubitags message dissemination we would create as many exchange points as ubisense tags. Interested parties to the ubitag events may include applications proving some automated control such as turning the lights on when a certain ubitag approaches or is inside a defined point/area within the localized area scanned by the Ubisense sensors.

[1] https://www.rabbitmq.com

D. Context-Aware Camera Control

In general, camera control can be easily realized through access to the camera's application programming interface. Many off-the-self IP cameras provide detailed information on how one may remotely control them through a web interface. Some cameras, though, provide access to a CGI based interface where one may control the camera without having to go directly to a browser.

The camera used for our prototype was a typical Foscam FI9831W (Fig. 5) costing about 100 Euro and features numerous functions to control movement in pan/tilt spaces through a simplified http CGI interface. For example, by calling the http CGI command:

Figure 5. FI9831W Network Camera.

"/cgi-bin/CGIProxy.fcgi?cmd=ptzMoveUp", the camera tilts up while calling "/cgi-bin/CGIProxy.fcgi?cmd= ptzMoveLeft" the camera turns left [10].

There have been numerous attempts to control camera movement by running face tracking algorithms onto current video frames and, thus, directing camera movement towards the detected object [11]. One of the drawbacks of these attempts is that the tutor needs to maintain a constant frontal line of view with the camera in order for correct face recognition. Our implementation with the Ubisense coordinate service does not depend on face recognition or silhouette tracking as real x,y,z coordinates are provided in real time by the existing infrastructure.

As it has been mentioned previously, Ubisense provides in real time the x,y,z coordinates of a ubitag attached to the tutors's body. In our case, we convert the ubitag's real world coordinates into screen coordinates in order to track the tutor's position. As seen in Fig. 6, the ubisense tracking system shows in real time how the object being tracked appears in Location Configuration panel. Once calibration has been set and defined coordinate points for Ubisense sensors (located at the four corners of the room), then the ubitag's coordinates are available in real time for further processing. For this example, we first set the sensor (as shown in lower right corner in Fig. 6) as the reference sensor (0,0,0) and the other sensors at predefined points after precise measurements. The whole calibration had the effect of creating a coordinate system where x increases moving towards left, y increases moving towards the upper side of the tracked area and z increases while moving upwards. Since the smartboard and the tutor were located between the 2 far sensors, we decided to drop the significance of the y coordinate and concentrate only on x and z. X coordinate would determine how much the camera will turn towards left or right while Z coordinate would represent the upwards or downward camera's movement.

As the video feed of the camera in pushed into the control unit responsible for adjusting the pan and tilt movement of the camera, there is a need for coordinate conversion through a simple normalization process: for each new screen coordinate derived from a corresponding ubisense coordinate the following normalization has to take place:

$$newVal = ((max'-min')/(max-min))*(ubiVal-min) + min'$$

where *max* and *min* represent the maximum and minimum ubisense coordinates and *max'* and *min'* are the new coordinates (screen coordinates). *ubiVal* represents the current ubisense coordinate delivered from the system (Fig. 7). Therefore, to convert Z ubisense coordinate to Y screen coordinate, we have to substitute *min* = 0 and *max* = "maximum available Z ubisense value". We chose *max* to be the value of 3 meters as a person carrying a ubitag cannot be more than 3 meters tall) and *min'* = 0 and *max'* = "maximum screen coordinate" which in our experiment is set to 764 (due to the 1024x764 screen resolution of the video feed). Alternatively, for X ubisense coordinate conversion, we chose *min* = 0 and *max* = 4 meters wide (room dimension) and *min'* = 0 and *max'* = 1024. The reason for the above computation is to translate the real world coordinates of the tag to screen coordinates so that we can compute the tag's point reference in the video frame and consequently control the camera based on the coordinates (screen) of the tag.

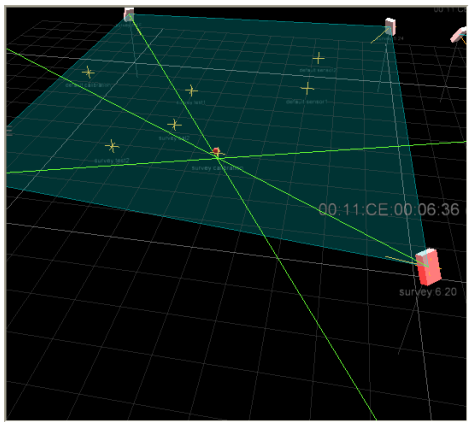

Figure 6. Ubisense's Location Configuration Control.

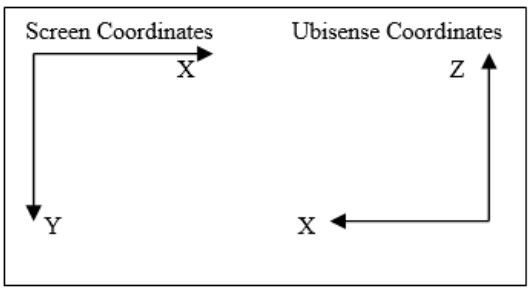

Figure 7. Screen coordinates vs. Ubisense coordinates.

Finally, once we have the screen coordinates computed, for each frame we get from the video camera, we defined a buffer zone within which no commands will be sent to the remote camera. That is, once the person/ubitag is within the boundaries of the square as we have defined in Fig. 8, there will be no camera motion.

Once the person moves away from the buffer zone (e.g. to the right), the control unit will sent an http request to the camera to turn right until the person is within the defined

boundary. Likewise if the person squats and the ubitag is below the boundary, the control unit will sent moveDown command to the camera. The algorithm that is used runs every 1.5 seconds in order to compensate for delays induced due to motor control start and stop sequences.

In Fig. 8, it is evident that the camera moves toward the ubisense tag held on the subject's hand. For this scene, we labeled the tag with a red dot while its screen coordinates show right below.

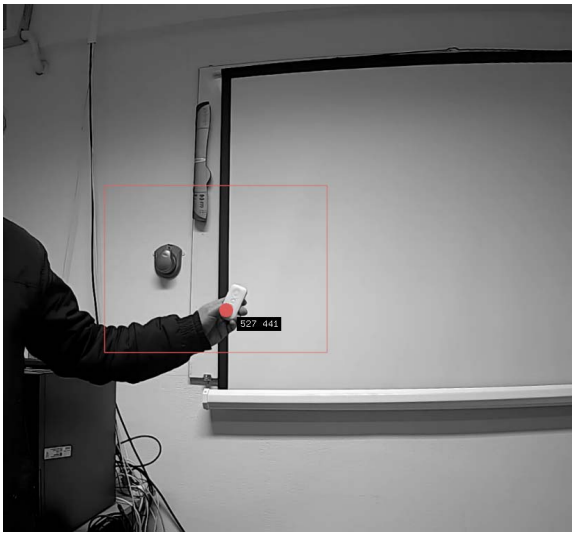

Figure 8. Camera stationed at screen coordinates 527x441.

By the term screen coordinates, we refer to the current video frame and how the pixels of the screen are arranged. For example, for a video frame of resolution of 1024x764, the point (0,0) would be at the top-left corner of the frame and (1024,764) at the bottom-right corner of the frame. That is, x increases to the right and y towards the bottom of the frame. As the subject moves towards the right side of the smartboard, the camera follows the ubitag and updates its position to a new set of screen coordinates at 573x391 (Fig. 9).

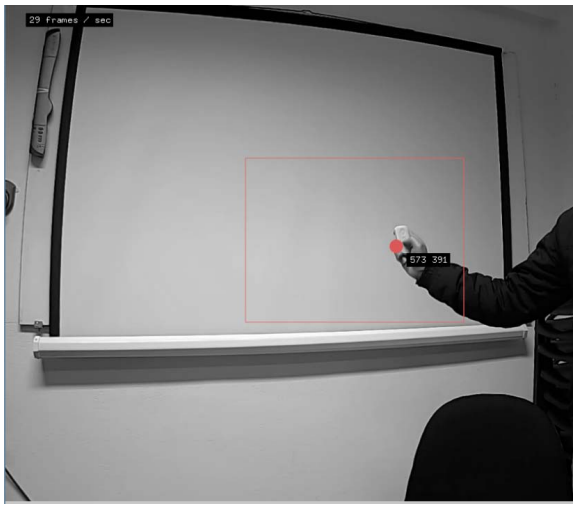

Figure 9. Camera moved to screen coordinates 573x391.

By examining the last two figures we can see that the camera has panned to the right from its initial position denoted in Fig. 8 Furthermore, if the person moves away from the screen, the camera will still track up to the point where it is physically possible for the camera to turn. For example, the camera FI9831W has a pan range (turn left/right) of about 300 degrees.

The control of the camera can be refined by combining location tracking with other sources of context. By monitoring the touch events on the smartboard we can infer when the tutor is writing on the board so that the view can be focused on that area. In addition, using Microsoft Kinect technology the system can infer whether the tutor is holding an object so that the camera can zoom on that object. The tutor by using a specific gesturing movement can signal the event of holding an object which instructs the camera control software to zoom at the position of the standing tutor [12].

E. Video Streaming

In this section we present the video streaming solution that we have developed for supporting the smart classroom application prototype using open source tools solely. We also present the way in which we create streaming events so as the end user be able to access them through Internet.

For video and audio streaming purposes we use the Red5 media streaming server. Red5 is an open source media streaming server implemented in Java that is able to transfer video, audio and generally any data in real-time over Internet between a server and its clients. Red5 was chosen among other solutions as best fitting our criteria such as pricing and open source. Alternatives are the Adobe Flash Media Server and Wowza Streaming Engine. Red5 supports many different video formats such as FLV, F4V, MP4, 3GP and audio formats such as MP3, F4A, M4A, AAC.

By using the Red5 Media Server, we are developing an open and extensible platform that can be used with our location tracking component for delivering the content of the classroom to remote students. Specifically the system must provide video/audio streaming from web cameras in online mode. The clients should be able to view video through the flash player embedded into a web page.

Our system's architecture consists on the use of IP Cameras for tutor's recording and live transmitting. The IP Camera uses the RTSP (real-time video streaming protocol) with H.264 support for streaming the video. Thus, our first concern is to create a stream from the network camera to our application. After that we need to convert the RTSP stream to RTMP format (the protocol used by adobe flash). To accomplish this we use FFMPEG tools [13] on our server. FFMPEG is a free library software for handling multimedia data and is licensed under the GNU Lesser General Public License (LGPL) version 2.1. FFMPEG includes libavcodec, an audio/video codec library, libavformat, an audio/video container mux and demux library, and the ffmpeg command line program for transcoding multimedia files. The RTSP to RTMP trans-coded live stream is in FLV (Flash Video) format. The end user can access the live streaming event simply by visiting the website URL in which the flash player is embedded. Fig. 10 shows the streaming end-to-end flow.

The quality of the streaming service depends mainly on the outbound bandwidth supported by the streaming server and its hardware set up. The bandwidth required depends on various parameters such as the maximum number of simultaneous users, the number of video/audio streams the

server can support per user, limitations of the server hardware and the target encoding of the audio/video streams. Following a one to many video on demand approach, the server bandwidth depends primarily on the number of simultaneous users and the average bitrate of encoded content, whereas the client bandwidth depends on the encoded content and the protocol overheads. Given that the audio/video produced by our camera has a codec rate of 2 Mbps (corresponding to a 1280x720p resolution and 30 frame-rate per second), then for 25 on line users, the server must have a transmission capability of at least 50 Mbps. On the client side, an ADSL line of a typical speed of 4 Mbps should be considered adequate. For our experiments, the maximum outbound bandwidth that our network could support was about 90 Mbps.

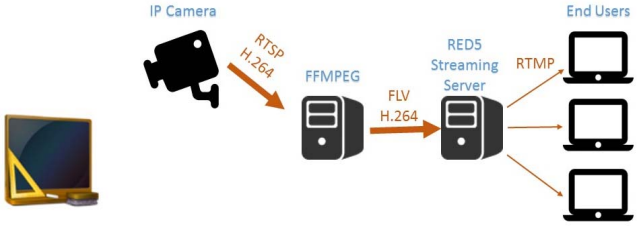

Figure 10. Streaming end-to-end flow.

F. *Preliminary Evaluation*

The main goal of this work was to create a multimedia experience for remote students and free the tutor from the complications of having to present the lecture within a constricted space (e.g. standing in front of a podium). Of course, the tutor still has to move within a predefined calibrated space but this new experience provides the tutor with a larger degree of freedom as the tutor may move freely and have the camera follow tutor's movements.

We presented the system to five tutors in HOU who have used extensively a virtual class system in the past for delivering courses to remote students. After the first demonstrations, almost all of the participating tutors said that this system is useful and can enhance the learning experience of remote students. They also underlined that there is a need to provide the solution in a way that integrates with the existing Content Management System (CMS), so that one may create video streams and share video streams in you-tube like way.

As far as the students were concerned, a set of 20 students participated in a test session of the system attending a lecture of 1h duration. Almost 85% of them reported good viewing conditions as all of them connected to the service from their homes using adsl network. 15% reported network glitches and these may be attributed to their overloaded phone network. 87% reported good or very good opinion on the prototype and that they would like to use it again.

We also asked students to provide their suggestions for improving the prototype. Almost all of them reported that they would like to save the recorded stream for later playback but this is an option being discussed to be included in a protected CMS solution. In addition, several students expressed their unease on the way they could interact throughout a lecture. Interaction between the tutor and the remote students or between local and remote students (e.g., raising hands for questions, participating in electronic polling/quizzes etc.) requires using HOU's teleconferencing platform for virtual classrooms. Our location-aware video streaming solution complements and enhances the interaction experience provided by the teleconference system, however the two solutions need to be integrated. This would enable remote students to see on their screen not only the lecture's slides but a live image of the classroom and listen to the sound from the classroom in real time.

Finally, since mobile support has become a priority we need on the one hand to tune the provision of the streaming service over cellular networks (e.g., 3G, 4G) and on the other hand to support adaptive streams to effectively serve heterogeneous devices and screen sizes. For example, our set of streams needs to be formatted for iOS using HTTP Live Streaming (HLS), instead of Flash.

IV. Related Work

Researchers in Tsinghua University at Beijing proposed a smart classroom model focused on tele-education [3],[14]. Amongst the various services offered the basic service the model proposed is called smart cameraman service and its purpose is to enable remote students to focus their attention on specific types of context. The remote student has only a viewport to the classroom from the camera. The proposed service assigns the appropriate camera from an array of cameras that are placed in the room according to the type of context recognized.

LOCAL project (Location and Context Aware Learning) utilizes location information and contextual information of students for enhancing the learning process in the context of a smart classroom [15]. The location system of LOCAL supports different types of methods for determining the position of a student who is obliged to carry some sort of mobile device. When the student is authenticated by the system the tracking and recording of the position of the mobile device starts. This recorded information in conjunction with data derived from students profile assists the learning process. It enables the tutor to send messages to specific learners or groups of learners according to their interests and goals they have set.

Scott and Benlamri [16] proposed a system that takes advantage of mobile technologies and ubiquitous computing and has been designed to help participants to communicate with each other during the educational process. The proposed system makes use of the location data of the user and relates him with nearby users (trainers and trainees) and learning objects. Using infrared technology (IR), it installs short-range communication networks among participants who are in range of the beacons visibility. This interaction partially follows the client/server model, as the need for existence of central server only is limited through the authentication process and network management.

The Taipei University proposed its own smart classroom standard where the main module which is called smart timetable consists of a timetable device which can be handled by the tutor [17]. Specifically, the timetable is able to track the position of the tutor in the area and by combining gesturing movements can operate accordingly. The area in front of the timetable is divided into several zones which have different corresponding operation flows.

Depending on the area the tutor stands, she can perform different actions. For example within the "touch zone" she can interact with the timetable and manage the room's components (lightning, air conditioning, etc.). Being within the "view zone" the smart timetable displays the information of the classroom weekly timetable. Finally within the "inform zone" after being authenticated by the system she is able to receive personal messages.

Alongside with the implementation of supportive systems associated with both the educational process and students communication either remotely or locally, research has been conducted regarding how ubiquitous computing can provide feedback mechanisms during the learning process. Such mechanisms can provide a measurement of the quality of lectures and educational process in general. Yang and Chien presented a research in which with the use of cameras and wireless technology they recorded the face and eyes of the students during the lecture in order to identify possible inattention which may be due to several factors [18]. By using PTZ cameras they could detect every student in the classroom and with the use of face and eye detection and recognition techniques they could determine his learning profile.

Our contribution with respect to the related work could be summarized as follows:

- Improvement of the accuracy of a commercial location-tracking system in order to meet our requirements.
- Provision of a context-aware camera control in order to provide a "live" image of the classroom to the remote students.
- Initial evaluation of our system that indicates the usefulness of our approach and informs further developments required.
- The overall low cost of the provided solution. Our system, except from the Ubisense, is based on open source technologies, thus eliminating the need to use expensive third party technologies. Furthermore, it enables capturing a lecture without the need to use extra support (e.g., a camera man).

V. CONCLUSIONS

This paper has given details of developing a context-aware video streaming solution in a smart classroom for enhancing distance learning experience, using a location tracking system with improved accuracy and Red5 media streaming server. Besides the work described in this paper other projects under development in the smart classroom include developing physical interaction services using Microsoft Kinect technology [12], speaker tracking based on microphone array and location sensing, collaborative learning applications based on the Android platform, etc.

The feedback from our preliminary evaluation indicates the need of integrating the developed system with other existing infrastructures such as Moodle and Saba Centra for delivering more advanced distance learning services to the end users. Another point of interest is the quality of the delivered data that has to be improved through pushing content to more robust content delivery services. There is also a need to provide an infrastructure to save and serve the recorded sessions on demand. Of course, systems like these exist (e.g. Kaltura) but there seem to induce a level of difficulty regarding integration that is repelling. Therefore, research is being conducted to develop an in-house media CMS system based on open-source technology.

ACKNOWLEDGMENT

The research described in this paper has been co-funded by the European Union (European Social Fund–ESF) and Greek national funds through the Operational Program "Education and Lifelong Learning" of the National Strategic Reference Framework (NSRF) (Funding Program: "HOU").

REFERENCES

[1] R.M. Bernard, B.R. Rubalcava and D. St-Pierre, "Collaborative online distance learning: Issues for future practices and research", Distance Education, vol. 21, pp. 260–277, 2000.

[2] Y. Beldarrain, "Distance education trends: integrating new technologies to foster student interaction and collaboration", Distance Education, vol. 27, pp. 139-153, 2006.

[3] Y. Shi, W. Xie, G. Xu, R. Shi, E. iChen, Y. Mao and F. Liu "The smart classroom: merging technologies for seamless tele-education", IEEE Pervasive Computing, vol. 2, pp. 47-55, 2003.

[4] J. Dooley, V. Callaghan, H. Hagras, M. Gardner, M. Ghanbari and D. Al-Ghazzawi, "The intelligent classroom: beyond four walls", Intelligent Environments, pp. 457-468, 2011.

[5] A. Varshavsky and S. Patel, "Location in Ubiquitous Computing, In Krumm, J. (ed.), Ubiquitous Computing Fundamentals, Taylor and Francis Group, Florida, Chapter 8, pp. 285-319, 2010.

[6] G. Deak, K. Curran and J. Condell, "A survey of active and passive indoor localisation systems", Comput. Commun. Vol. 35, pp.1939-1954, September 2012.

[7] P. Steggles and S. Gschwind, The Ubisense smart space platform", Adjunct Proceedings of the Third International Conference on Pervasive Computing, vol. 191, pp. 73-76, 2005.

[8] N.J. Gordon, "Bayesian methods for tracking", Imperial College, University of London, London, 1994.

[9] K. Schwarz, "Darts, dice, and coins: Sampling from a discrete distribution" Dec 2011. Available: http://www.keithschwarz.com/darts-dice-coins/

[10] CGI Interface of Foscam Camera Control. Available: http://foscam.us/forum/cgi-api-sdk-for-mjpeg-camera-t2986.html#p13630

[11] C. Yang, R. Chen, C. Lee, and S. Lin, "PTZ camera based position tracking in IP-surveillance system", 3rd International Conference on Sensing Technology, pp. 142-146, November 2008.

[12] C. Charachristos, C. Goumopoulos and A. Kameas, "Enhancing collaborative learning and management tasks through pervasive computing technologies", Ambient Computing, Applications, Services and Technologies, pp. 27-33, August 2014.

[13] FFmpeg, A complete, cross-platform solution to record, convert and stream audio and video. Available: http://www.ffmpeg.org

[14] X. Li, L. Feng, L. Zhou and Y. Shi, "Learning in an ambient intelligent world: enabling technologies and practices", IEEE Trans. Knowl. & Data Eng., vol. 21, no. 6, pp. 910-924, June 2009.

[15] J. Barbosa, R. Hahn, S. Rabello, and D. Barbosa, "LOCAL: A model geared towards ubiquitous learning", ACM SIGCSE Bull., vol. 40, no. 1, pp. 432-436, 2008.

[16] K. Scott and R. Benlamri, "Context-aware services for smart learning spaces", IEEE Trans. Learn. Technol., vol. 3, no. 3, pp. 214-227, Jul./Sep. 2010.

[17] Y. Yu, S. D. You and D. Tsai, "Smart timetable plate for classroom", Advanced Learning Technologies, pp. 552-554, July 2010.

[18] Yang, S. and Chien, L. "A face and eye detection based feedback system for smart classroom", Electronic and Mechanical Engineering and Information Technology, pp. 571-574, 2011.