DroneCAST: Towards a Programming Toolkit for Airborne Multimedia Display Applications

Ragavendra Lingamaneni
Stuttgart Media University
Department of Electronic Media
Nobelstrasse 10, 70569
Stuttgart, Germany
lingamaneni@hdm-stuttgart.de

Jürgen Scheible
Stuttgart Media University
Department of Electronic Media
Nobelstrasse 10, 70569
Stuttgart, Germany
scheible@hdm-stuttgart.de

Thomas Kubitza
University of Stuttgart
Institute for Visualization and
Interactive Systems (VIS)
Pfaffenwaldring 5a, 70569
Stuttgart, Germany
thomas.kubitza@vis.uni-stuttgart.de

MobileHCI '17, September 04-07, 2017, Vienna, Austria
ACM 978-1-4503-5075-4/17/09.
https://doi.org/10.1145/3098279.3122128

Abstract
In recent years, new type of public displays have captured the interest of researchers-displays with the ability to move freely in three-dimensional space. We refer to such display systems as Airborne Multimedia Display (AMD) systems. In this paper, we provide a comprehensive analysis of requirements for developing interactive AMD applications based on extensive literature survey of the related work and from our own experience of flying AMD systems. We then outline the design and implementation of DroneCAST: a programming toolkit for developing AMD applications with remote delivery and control mechanism for multimedia content with IoT middleware as the core part of the system. Finally, we build a sample AMD application with the toolkit to further illustrate the applicability of the system.

Author Keywords
Airborne Displays; Requirements Analysis; Toolkit; Drones; Interaction

ACM Classification Keywords
H.5 [Information interfaces and presentation]: User Interfaces

Introduction
Digital displays are increasingly permeating public and semi public places. The ability to change their content quickly

Figure 1: (from left to right)1. In-situ display drone 2. Display drone with attached projector 3. Mid-air display concept prototype

is responsible for replacing traditional paper posters and other permanent static signs with digital displays. However, almost all such display systems are static in nature i.e., they are either installed permanently or their position is fixed during the event.

Soon, we envision new types of flying or hovering aerial display systems, which can move freely in three-dimensional space. We call such display systems as Airborne Multimedia Display (AMD) systems.

Airborne displays in the form of kites and blimps with brand signs, small airplanes towing banners and skywriting existed since very long. However, the content displayed is typically static and non-interactive. In recent years, research work like *DisplayDrone*[9], *In-situ DisplayDrone* [8], *Midair Displays* [11] show that AMDs with dynamic and interactive display content are indeed feasible and should be explored further. A step in this direction is to research on building interactive multimedia applications for AMDs. In this paper we introduce DroneCAST, a toolkit for building AMD applications. The aim of this toolkit is to support developers in programming and deploying interactive multimedia applications for AMD systems. DroneCAST provides abstraction over the complexities of AMD systems and offers a simple approach for application developers to create applications with HTML, CSS and Javascript like traditional web applications.

Related Work

Various prototypes of AMD systems are built in the recent past. One such novel system was introduced by Scheible et al. [9] called *DisplayDrone* that combines a rotor copter with a video projector and a mobile phone which can be used to project multimedia content onto walls and arbitrary objects in the physical space. Another display system using a rotor copter was presented by Schneegass et al. where they introduced the concept of *Midair Display* [11]. It consists of a 10" display mounted on the rotor copter. Nozaki presented *Flying Display* [4] that consists of two UAVs (Unmanned Aerial Vehicles). One with a projector following the other carrying a screen. Similarly, Oh et al. presented an autonomous blimp with a visual tracking system and an image projection system [5] where the real-time visual system tracks the blimp while it flies along a given spatial path to

follow a wall and the image projection. Here, the system projects still images or a video stream onto the blimp surface.

BitDrones [7] take a different approach by building interactive real reality 3D displays that use nano-quadcopters as self-levitating tangible building blocks. This platform uses indoor tracking technology to track the drones.

To build applications for interactive public displays there are numerous toolkits available that offer a means for application developers. PuReWidgets [1] toolkit supports multiple interaction mechanisms, context acquisition as well as user interface creation. SenScreen [10] toolkit eases up sensor integration on public displays by providing low-level connection to sensors and provides the data via an API that allows developers to write their applications in JavaScript.

Application development for AMD systems is hardly explored before in detail with few exceptions. Scheible et al. introduced *DisplayDrone* [9], which had an SMS application built on the phone which projected SMS text sent by the viewers on arbitrary surfaces, thus turning it into an interactive system. However, it did not have any other multimedia content delivery mechanism. In an other example, Blimps were used as floating avatar in telepresence applications [12].

Requirements Analysis

In this section, we present the requirements analysis for AMD systems. To extract these requirements, we performed an extensive literature survey about interactive mobile display systems by searching online databases (such as ACM, IEEE, Google Scholar) and filtering publications by keywords such as “display drone“, “flying display“, and “interactive display“, from the last 10 years.

Apart from the above literature survey, from our own experience in flying AMD systems as mentioned in [9] and [8] we identified the areas that can be addressed in the proposed toolkit to help ease application development process for AMD systems. While some of these requirements are common to public display systems, most of them are specific to airborne displays. Here we present the areas that we consider relevant.

Programmable Dynamic Content Update
Unlike static public display systems, which are usually fixed at certain locations, AMD systems are usually designed to fly from one place to another or stay airborne. In many scenarios of AMD deployment, updating the content manually or interactively on the displays may not be feasible, as they are not reachable or may be out of visible sight. Application programmers, however, should be able to change the content on the display by acquiring context from various sensors like Altitude, proximity, GPS, BLE scanner etc. Hence, our toolkit should provide support for context acquisition and a logic description interface for dynamic content update.

Dual Device Interaction
Muller et al. explored interaction modalities [3] in their design space analysis of interactive public displays. However, most of the discussed interaction methods cannot be applied one to one for AMD systems as these may be in far distance from the user compared to traditional static displays. In addition, AMD systems may be in continuous motion most of the time.

For input modality, we mainly identify *Remote Control* method as main form of interactivity. Current audience-to-display interaction approaches are often based on mobile phones, which connect to the display. With such an arrangement, it should be easy for the viewers to share media content with

the AMD even when at a distance. AMD systems should support single user interaction as well as group interaction. Such interactions are employed when developing applications like games or social applications which are deployed at events in example for entertaining crowds. We call this approach as dual device interaction [6].

Interaction Initiation Mechanisms
When using the *Remote Control* approach for interaction, connecting to an AMD is a major challenge as users don't have any idea how to start interacting with it. To develop interactive AMD applications, potential AMD system users need to connect, interact and engage with the system. The toolkit should therefore provide an easy mechanisms for the users to connect to the system.

From the technology perspective, connection can be realized via an Internet connection between remote controller (AMD system user) and the AMD. However, communicating a URL address to the user is not straightforward as traditional methods like NFC chip scanning or via Bluetooth is not feasible due to the moving of the AMD. QR code scanning for URL can be used in cases when the AMD is near by the user for scanning. Other methods like WiFi hotspot is more suitable, but limited for a small sized area as long as the hotspot is available.

Persistence Channels
Due to the limited form of interaction with AMD systems, additional channels should be provided to sustain the interaction even after the AMD is out of visible range. We call such channels *persistent channels*. We believe persistence channel is very important when interacting with AMD systems. Through this channel users are engaged even after the display is out of the field of view. Persistence channels can be in the form of SMS, Email or just a HTTP connection.

Support for Flight Characteristics
The main differentiator when comparing regular fixed position public display with an airborne public display is its ability to fly. Various properties of airborne displays such as speed, direction and altitude have never been explored before in context to public displays. Providing access to these properties through various sensor values will help application developers design and develop unique application for AMD systems. Other characteristics such as six degrees of freedom and spatial localization can be exploited to build applications that employ multiple AMD systems and act in tandem to create 3D displays in space as explored in *BitDrones* research project [7].

Navigation
Both autonomous and manual navigation can be used for the AMD system. In case of manual navigation, a pilot is steering the UAV either from a remote location or staying on site. Whereas in autonomous navigation, the path is fixed and the UAV is guided along the set path through various tracking technologies. The type of navigation can be an important dimension in the requirements of the AMD as either of them can be selected based on the context of use.

Factors Affecting the Flight
Among the external factors that can effect the AMD, weather plays a very important role when deployed in outdoor scenario. Using real-time weather information in the applications can be a huge factor in avoiding failures and inconvenience. In some countries there are legal regulations to what extent UAVs can be used for commercial purposes. Such information can be made available for application developers for safe and legal deployment of AMD systems.

Display Abstraction
We envision that AMD systems of many different types, forms and sizes will be built. However, the core component

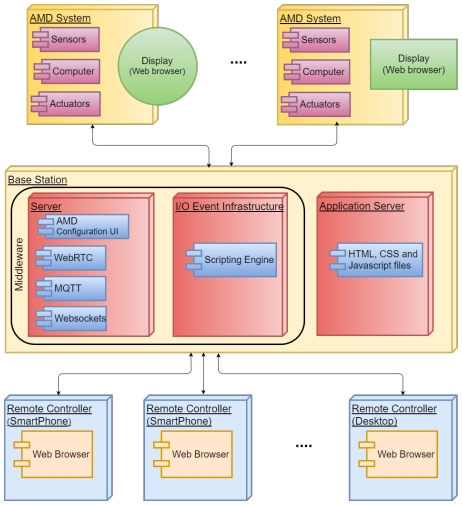

Figure 2: Overview of the DroneCAST toolkit architecture.

of an AMD system is its display technology. Many types of display technologies are currently commercially available. Some of the interesting ones among them are light weight flexible displays, transparent displays and laser projectors. Providing abstraction from the display technologies that are suitable for using in AMD systems will reduce the application development time.

DroneCAST System Concept & Implementation

In this section, we will describe the architecture and main components of the DroneCAST programming toolkit. The main goal of this toolkit is to enable developers to develop AMD applications with regular web technologies. The toolkit consists of four components: base station running the IoT middleware, Remote controller component used on the smartphone or tablet, AMD system and an optional network infrastructure. An overview of the platform is depicted in Figure 2

Base Station

The Base station is the central part of the toolkit and runs the DroneCAST IoT middleware component called meSchup [2]. meSchup middleware was originally developed for programming interaction behaviors in smart environments. The middleware consists of a I/O infrastructure which acts as a sink for all the I/O events triggered by the AMD systems or the remote controllers. The middleware provides an abstraction over several types of sensor data and provides uniform access. A web based JavaScript editor is used to script the routing of these events as necessary by the application. The Base station is equipped with communication adapters for technologies like cellular data communication, Wi-Fi, Bluetooth and BLE.

AMD System Component

An AMD can be built using different types of UAVs, display technologies and a computer with additional sensors/actuators. Our DroneCAST middleware provides an AMD configuration GUI for building the system as per the requirements and provides abstraction from all the underlying components and presents the system as a single unit at the application development layer.

Remote Controller Component

Remote controller is a web application running in the web browser of a smartphone or a desktop. Additionally, it can be a native smartphone application that supports webviews. Currently we provide a JavaScript library for the application developers to add in their remote controller applications. This library is responsible for connecting to the DroneCAST middleware. Once the remote controller register to the server asking to connect to a AMD application,

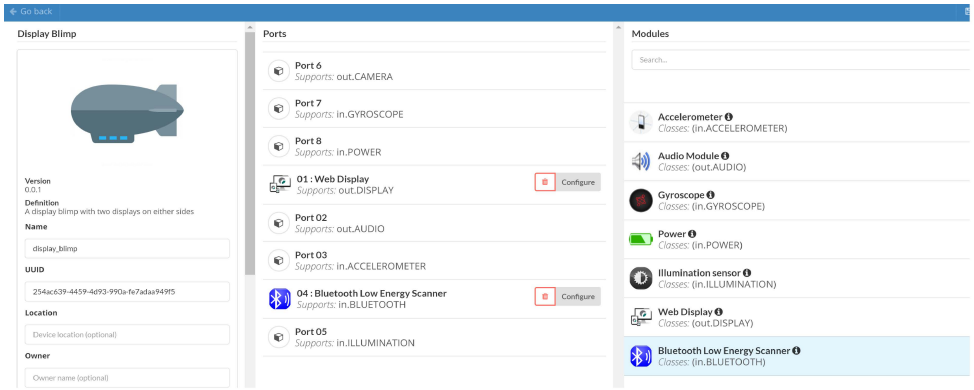

Figure 3: AMD configuration GUI. The left most column contains the information about the drone. The right most column lists all the modules that are available for the selected drone. The middle column contains the ports to which the modules can be assigned.

the server responds by sending the appropriate user interface files related to the application.

Network Infrastructure

An optional network infrastructure is needed depending on the type of connectivity employed by the AMD deployment. By default, DroneCAST works on the cellular data (3G/4G/LTE) connection. However, when local deployment is preferred, an additional network infrastructure (e.g. Wi-Fi access points) is used with base station as the central component.

Sample Application : Flying Display Blimp for Indoor Exhibition Hall

In this section, we will develop a sample application using the toolkit that potentially could be deployed as real AMD application and get the first impressions through hands-on experience.

Scenario: A small sized display blimp with a lightweight flexible display attached on both of the wider sides, is flying in a large exhibition hall. The flight is controlled manually by a controller. When the blimp is above a certain booth, it will display relevant information about that booth for the exhibition visitors to see. Each booth is identified by a Bluetooth Low Energy (BLE) beacon so that each beacon is associated with a particular media content. In the configuration GUI, a blimp based AMD system is configured with a webview and a BLE scanner. In the scripting engine, scripts are written in a simple if-then format such that when a particular BLE MAC ID is detected the corresponding media is shown on the display. In this application, remote controllers are however, not involved and the display content is changed only based on the script logic.

Explanation

In this section, we will walk-through how the above scenario is built using the toolkit. Firstly, the required display blimp is chosen in the toolkit and configured as show in the figure 3. Among various modules available for this particular blimp we only configured it with a web-display and a BLE scanner. Web-display is used to display all the HTML content and the BLE scanner on the blimp is used for scanning the beacons and return the beacon with the largest RSSI (Received Signal Strength Indicator) value every second. After configuring the blimp with required modules, interaction between the display and the BLE scanner is scripted using the script editor of the toolkit as shown in the figure 4. The script is executed whenever a sensor input from the BLE scanner is sent to the middleware. For example, if the BLE beacon mac id of booth 1 in the exhibition hall is “de6b23dff886", when the blimp passes over booth 1, the BLE scanner detects the beacon with mac id “de6b23dff886" and hence according to the script webpage for booth 1 “http://127.0.0.1:8080/booth1/advt.html" is displayed on both the sides of the blimp display.

Conclusion and Future Work

In this paper, after extensive literature survey of the related work, we have derived the system requirements analysis for interactive Airborne Multimedia Displays. We built a platform called DroneCAST to ease developing applications for such displays. We used web based platform to take advantage of the familiarity of web application development. An existing IoT middleware was used as the core of the toolkit after the requirements analysis. A sample application was built to show the easy with which AMD applications can be developed with the toolkit. In the future, we intend to evaluate the toolkit along with various interaction techniques, UI paradigms and various applications for multiple deployment scenarios and flying AMD systems in the true environment

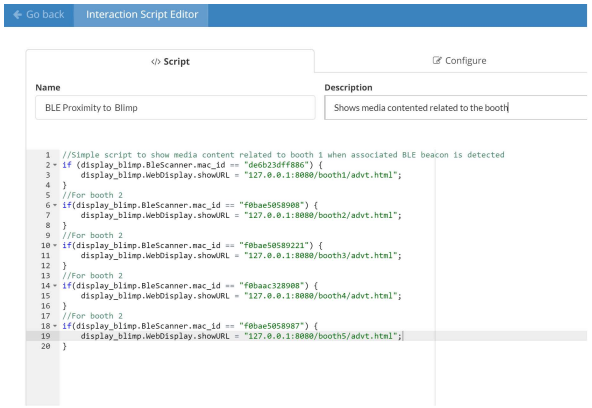

Figure 4: Interaction scripting editor of the toolkit

of use. We deliberately left out discussing security related issues as this is a whole topic on its own to explore and describe sufficiently here. The DroneCAST toolkit is in active development.

Acknowledgments

The research leading to these results has received funding from the MFG Stiftung under Karl-Steinbuch-Forschungsprogram for DisDrone project.

REFERENCES

1. Jorge Cardoso and Rui José. 2012. PuReWidgets: A Programming Toolkit for Interactive Public Display Applications. In *Proceedings of the 4th ACM SIGCHI Symposium on Engineering Interactive Computing Systems (EICS '12)*. ACM, New York, NY, USA, 51–60. DOI:http://dx.doi.org/10.1145/2305484.2305496
2. Thomas Kubitza and Albrecht Schmidt. 2015. *Towards a Toolkit for the Rapid Creation of Smart Environments*.

Springer International Publishing, Cham, 230–235. DOI: http://dx.doi.org/10.1007/978-3-319-18425-8_21

3. Jörg Müller, Florian Alt, Daniel Michelis, and Albrecht Schmidt. 2010. Requirements and Design Space for Interactive Public Displays. In *Proceedings of the 18th ACM International Conference on Multimedia (MM '10)*. ACM, New York, NY, USA, 1285–1294. DOI: http://dx.doi.org/10.1145/1873951.1874203

4. Hiroki Nozaki. 2014. Flying Display: A Movable Display Pairing Projector and Screen in the Air. In *CHI '14 Extended Abstracts on Human Factors in Computing Systems (CHI EA '14)*. ACM, New York, NY, USA, 909–914. DOI: http://dx.doi.org/10.1145/2559206.2579410

5. S. Oh, S. Kang, K. Lee, S. Ahn, and E. Kim. 2006. Flying Display: Autonomous Blimp with Real-Time Visual Tracking and Image Projection. In *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*. 131–136. DOI: http://dx.doi.org/10.1109/IROS.2006.281919

6. Scott Robertson, Cathleen Wharton, Catherine Ashworth, and Marita Franzke. 1996. Dual device user interface design: PDAs and interactive television. In *Proceedings of the SIGCHI conference on Human factors in computing systems*. ACM, 79–86.

7. Calvin Rubens, Sean Braley, Antonio Gomes, Daniel Goc, Xujing Zhang, Juan Pablo Carrascal, and Roel Vertegaal. 2015. BitDrones: Towards Levitating Programmable Matter Using Interactive 3D Quadcopter Displays. In *Adjunct Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology (UIST '15 Adjunct)*. ACM, New York, NY, USA, 57–58. DOI: http://dx.doi.org/10.1145/2815585.2817810

8. Jürgen Scheible and Markus Funk. 2016. In-situ-displaydrone: Facilitating Co-located Interactive Experiences via a Flying Screen. In *Proceedings of the 5th ACM International Symposium on Pervasive Displays (PerDis '16)*. ACM, New York, NY, USA, 251–252. DOI: http://dx.doi.org/10.1145/2914920.2940334

9. Jürgen Scheible, Achim Hoth, Julian Saal, and Haifeng Su. 2013. Displaydrone: A Flying Robot Based Interactive Display. In *Proceedings of the 2Nd ACM International Symposium on Pervasive Displays (PerDis '13)*. ACM, 49–54. DOI: http://dx.doi.org/10.1145/2491568.2491580

10. Stefan Schneegass and Florian Alt. 2014. SenScreen: A Toolkit for Supporting Sensor-enabled Multi-Display Networks. In *Proceedings of The International Symposium on Pervasive Displays (PerDis '14)*. ACM, New York, NY, USA, Article 92, 6 pages. DOI: http://dx.doi.org/10.1145/2611009.2611017

11. Stefan Schneegass, Florian Alt, Jürgen Scheible, Albrecht Schmidt, and Haifeng Su. 2014. Midair Displays: Exploring the Concept of Free-floating Public Displays. In *CHI '14 Extended Abstracts on Human Factors in Computing Systems (CHI EA '14)*. ACM, New York, NY, USA, 2035–2040. DOI: http://dx.doi.org/10.1145/2559206.2581190

12. Hiroaki Tobita, Shigeaki Maruyama, and Takuya Kuji. 2011. Floating Avatar: Blimp-based Telepresence System for Communication and Entertainment. In *ACM SIGGRAPH 2011 Emerging Technologies (SIGGRAPH '11)*. ACM, New York, NY, USA, Article 4, 1 pages. DOI:http://dx.doi.org/10.1145/2048259.2048263