Towards a Component-Based Platform for Industrial AR

Tim Verbelen
Ghent University - iMinds
Gaston Crommenlaan 8/201
B-9050 Gent, Belgium
tim.verbelen@intec.ugent.be

Pieter Simoens
Ghent University College
Valentin Vaerwyckweg 1
B-9000 Gent, Belgium
pieter.simoens@hogent.be

Bart Dhoedt
Ghent University - iMinds
Gaston Crommenlaan 8/201
B-9050 Gent, Belgium
bart.dhoedt@intec.ugent.be

UbiComp '13 Adjunct, Sept 8-12, 2013, Zurich, Switzerland.
ACM 978-1-4503-2139-6/13/09...$15.00.

Abstract
The roll-out of AR solutions in industrial environments goes beyond technical requirements and involves challenges regarding software deployment, management and maintenance. In this paper we present a lightweight runtime environment for AR applications, using a component-based management platform providing easy deployment, updates and reuse of software components.

Author Keywords
Software Components, Augmented Reality, OSGi

ACM Classification Keywords
H.5.1 [Multimedia Information Systems]: Artificial, augmented, and virtual realities.

Introduction
Augmented Reality (AR) is a promising driver for industrial applications. Already in the early nineties, AR technology was being considered at Boeing to aid factory workers in manufacturing airplanes [6]. Nowadays AR is still a hot research topic for industrial applications, for example using head-up devices for assistance in automotive assembly [5], order picking [11], etc. Despite continuous advances in hardware miniaturization, wearable displays, tracking technologies, etc. many challenges still remain in an industrial setting. For example, adopting an

AR order picking solution within a large company, results in a large amount of head-up display devices, that have to be manually installed, configured and maintained over time. However, current AR platforms such as Metaio's Unifeye [1], Artoolworks' ARToolkit [2], Qualcomm's Vuforia [3] only offer an SDK, but no dedicated runtime environment. Also, the solutions at hand are offered as monolithic libraries difficult to integrate with other code.

In this paper, we present two contributions to leverage industrial AR. First, we present a lightweight runtime environment based on the Linux kernel, avoiding common operating system services that are not used in a dedicated AR device, such as a window manager. Second, we propose a component-based management platform that takes care of runtime management of applications, enabling easy deployment, updating and reusing of software components.

Platform Architecture

The overall architecture of our platform is presented in Figure 1. As the platform has to run on mobile, embedded devices, we chose to build it from scratch based on a bare Linux kernel, using the Buildroot build system, which is based on the uClibC, a C library for embedded devices [4]. Together with the kernel we provide some core native libraries required for AR applications such as:

BusyBox a collection of many common UNIX utilities for embedded systems.

libDRM library that directly interfaces with the video hardware using the Linux kernel's Direct Rendering Manager (DRM).

Mesa3D an implementation of the OpenGL (ES) APIs for rendering 3D to the video hardware.

v4l2 library for interfacing with camera devices, enabling to capture video frames.

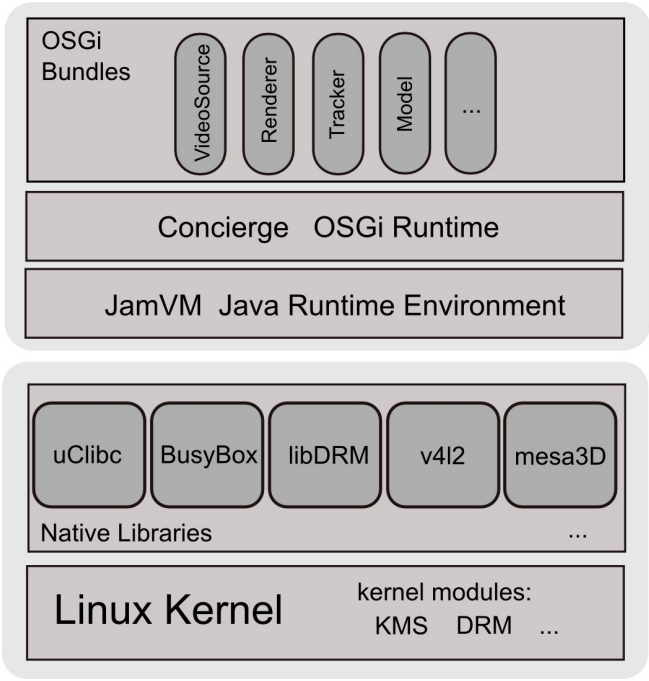

Figure 1: Overview of the platform: on top of a minimal Linux platform we deploy the OSGi service component runtime, providing reusable software components for AR applications.

In order to facilitate runtime management of applications, we adopted the OSGi service platform [9], a module system for Java that implements a complete and dynamic component model. Application components, called bundles, can be remotely installed, started, stopped, updated, and uninstalled without requiring a reboot, and dependencies between bundles are resolved and handled at runtime. OSGi incorporates semantic versioning, which

facilitates the management of software updates.

In order to support the OSGi module system, a Java runtime environment is required, as well as an implementation of the OSGi specification:

JamVM a lightweight Java runtime environment with a very small footprint and support for various architectures.

Concierge a highly optimized OSGi runtime targetted for embedded devices.

On top of the OSGi runtime then application specific bundles can be deployed. Many of these components can be reused in different applications, such as components for accessing hardware, rendering, tracking algorithms, etc. The OSGi platform can be used in conjunction with other underlying systems, for example a fully fledged Linux distribution or an Android kernel with a Dalvik VM. In the next section we will discuss an example component-based AR application.

Component-based AR Applications

We implemented a sample AR application on top of our platform, based on the Parallel Tracking and Mapping (PTAM) algorithm proposed by Klein et al. [7]. In order not to sacrifice performance, many of the computer vision algorithms are still implemented in optimized native code, which is accessed from within the OSGi bundles using the Java Native Interface (JNI). We identified the following key components, as illustrated in Figure 2:

VideoSource The VideoSource fetches video frames from the camera hardware. These frames are analyzed by the Tracker, and rendered with an augmented reality overlay by the Renderer.

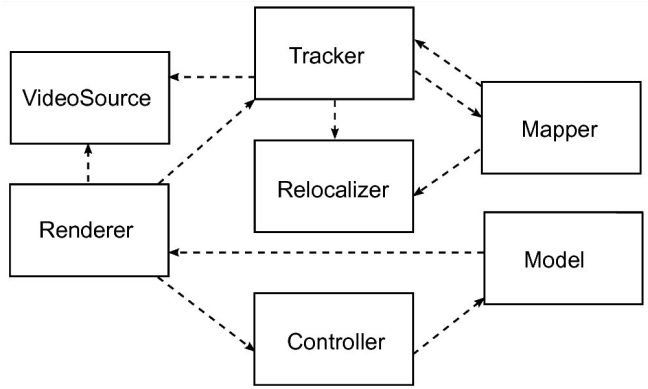

Figure 2: Overview of PTAM components.

Tracker The Tracker searches map points in video frames and calculates the new camera pose. When a new keyframe should be added to the map, this keyframe is sent to the Mapper.

Relocalizer When not enough map points are found in the video frame, the Tracker calls the Relocalizer, that estimates the camera position using the small blurry image approach.

Mapper The Mapper receives keyframes from the Tracker, that are used to initialize and extend the map. The Mapper also performs the bundle adjustment to optimize the map points. The Tracker and Relocalizer receive notifications when the map is updated.

Model Contains the model of possibly virtual items that have to be displayed as an overlay.

Controller Handles user input and updates the Model.

Renderer Each camera frame is rendered on screen together with an overlay of 3D objects specified in the Model. These 3D objects are aligned according to the camera pose given by the Tracker.

Adopting the component-based approach offers a lot of advantages over a monolithic implementation. First, software components can be automatically fetched over the network and installed on all devices usign the OSGi Bundle Repository (OBR) [8]. Second, this approach enables easy component upgrades, and even the implementation of components can be changed, i.e. the PTAM Tracker component can easily be exchanged with a marker based tracker without changing other application functionality. Finally, the framework can be used for offloading components to server infrastructure as proposed in [10].

Prototype implementation

We have a prototype implementation of the proposed platform running on an embedded PC equipped with an Intel Atom D525 dual core CPU clocked at 1.8 GHz. To asses the overhead of the Java platform we compared the cost of a method call in Java via JNI versus a pure C implementation. For each JNI call an extra cost of 120 ns has to be taken into account. However, this cost is very small as processing one camera frame can easily take up to 50 ms.

Conclusion

In this paper we presented a lightweight runtime environment and OSGi-based management framework for industrial AR applications. The component-based approach enables efficient deployment, updating and reuse of application components. First experimental results show that the overhead of the component layer is negligible compared to frame processing times.

References

[1] www.metaio.com/products/.

[2] www.artoolworks.com/.

[3] www.qualcomm.com/solutions/augmented-reality.

[4] Andersen, E. Buildroot: making embedded linux easy, 2005.

[5] Burgy, C., Garrett, J., Klausner Jr., M., J., A., and Nobis, G. Speech-controlled wearable computers for automotive shop workers. In *Proceedings of SAE 2001 World Congress* (2001).

[6] Caudell, T., and Mizell, D. Augmented reality: an application of heads-up display technology to manual manufacturing processes. In *Proceedings of the 25th International Conference on System Sciences* (1992), 659–669.

[7] Klein, G., and Murray, D. Parallel tracking and mapping for small ar workspaces. In *Proceedings of the 6th International Symposium on Mixed and Augmented Reality*, ISMAR '07 (2007), 1–10.

[8] The OSGi Alliance. RFC 112: Bundle Repository, 2006.

[9] The OSGi Alliance. *OSGi Service Platform, Core Specification, Release 4, Version 4.2*. September 2009.

[10] Verbelen, T., Simoens, P., De Turck, F., and Dhoedt, B. Cloudlets: bringing the cloud to the mobile user. In *Proceedings of the third ACM workshop on Mobile cloud computing and services* (2012), 29–36.

[11] Weaver, K. A., Baumann, H., Starner, T., Iben, H., and Lawo, M. An empirical task analysis of warehouse order picking using head-mounted displays. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (2010), 1695–1704.