Fast and Scalable Structure-from-Motion for High-precision Mobile Augmented Reality Systems

Hyojoon Bae

Dissertation submitted to the faculty of
the Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy
In
Computer Engineering

C. Jules White, Chair
Mani Golparvar-Fard
Jeffrey H. Reed
Peter H. Athanas
T. Charles Clancy

March 26th, 2014
Blacksburg, Virginia

Keywords: 3D Reconstruction, Structure-from-Motion, 3D Cyber-physical Modeling, Direct 2D-to-3D Matching, Image-based Localization, Mobile Augmented Reality

Fast and Scalable Structure-from-Motion for High-precision Mobile Augmented Reality Systems

Hyojoon Bae

ABSTRACT

A key problem in mobile computing is providing people access to necessary cyber-information associated with their surrounding physical objects. Mobile augmented reality is one of the emerging techniques that address this key problem by allowing users to see the cyber-information associated with real-world physical objects by overlaying that cyber-information on the physical objects' imagery. As a consequence, many mobile augmented reality approaches have been proposed to identify and visualize relevant cyber-information on users' mobile devices by intelligently interpreting users' positions and orientations in 3D and their associated surroundings. However, existing approaches for mobile augmented reality primarily rely on Radio Frequency (RF) based location tracking technologies (e.g., Global Positioning Systems or Wireless Local Area Networks), which typically do not provide sufficient precision in RF-denied areas or require additional hardware and custom mobile devices.

To remove the dependency on external location tracking technologies, this dissertation presents a new vision-based context-aware approach for mobile augmented reality that allows users to query and access semantically-rich 3D cyber-information related to real-world physical objects and see it precisely overlaid on top of imagery of the associated

physical objects. The approach does not require any RF-based location tracking modules, external hardware attachments on the mobile devices, and/or optical/fiducial markers for localizing a user's position. Rather, the user's 3D location and orientation are automatically and purely derived by comparing images from the user's mobile device to a 3D point cloud model generated from a set of pre-collected photographs.

A further challenge of mobile augmented reality is creating 3D cyber-information and associating it with real-world physical objects, especially using the limited 2D user interfaces in standard mobile devices. To address this challenge, this research provides a new image-based 3D cyber-physical content authoring method designed specifically for the limited screen sizes and capabilities of commodity mobile devices. This new approach does not only provide a method for creating 3D cyber-information with standard mobile devices, but also provides an automatic association of user-driven cyber-information with real-world physical objects in 3D.

Finally, a key challenge of scalability for mobile augmented reality is addressed in this dissertation. In general, mobile augmented reality is required to work regardless of users' location and environment, in terms of physical scale, such as size of objects, and in terms of cyber-information scale, such as total number of cyber-information entities associated with physical objects. However, many existing approaches for mobile augmented reality have mainly tested their approaches on limited real-world use-cases and have challenges in scaling their approaches. By designing fast direct 2D-to-3D matching algorithms for

localization, as well as applying caching scheme, the proposed research consistently supports near real-time localization and information association regardless of users' location, size of physical objects, and number of cyber-physical information items.

To realize all of these research objectives, five research methods are developed and validated: 1) Hybrid 4-Dimensional Augmented Reality (HD^4AR), 2) Plane transformation based 3D cyber-physical content authoring from a single 2D image, 3) Cached k-d tree generation for fast direct 2D-to-3D matching, 4) double-stage matching algorithm with a single indexed k-d tree, and 5) K-means Clustering of 3D physical models with geo-information. After discussing each solution with technical details, the perceived benefits and limitations of the research are discussed with validation results.

Acknowledgement

First of all, I would like to express my special appreciation and thanks to my advisors Professor Dr. Jules White and Professor Dr. Mani Golparvar-Fard, you have been tremendous mentors for me. I would like to thank you for encouraging my research and for allowing me to grow as a research scientist. Your advice on both research as well as on my career have been priceless.

I would also like to thank my committee members, Professor Dr. Jeff Reed, Professor Dr. Peter Athanas, and Professor Dr. Charles Clancy for serving as my committee members. I also want to thank you for letting my defense be a valuable and enjoyable moment, and for your brilliant comments and suggestions, thanks to you.

A special thanks to my family. Words cannot express how grateful I am to my mother and father for all of the sacrifices that you've made on my behalf. Your prayer and supports for me were what sustained me thus far. I would also like to thank all of my friends, especially Hamilton Turner, Thaddeus Czauski, Seunghwan Kim, Seungmo Kim, Jaeeung Kim, and Aram Lee, who supported me in researching and writing, and incented me to strive towards my goal.

Finally, I would like to acknowledge the financial support I received at Virginia Tech from the National Science Foundation and from graduate research assistantships.

Contents

List of Figures

List of Tables

1 Introduction

Automated, inexpensive, and fast access to surrounding cyber-information associated with real-world physical objects in the field has significant potential to improve real-world tasks, such as decision-making during construction or facility management activities. For example, fast access to construction cyber-information, which is usually in form of specifications, drawings, or schedule information, can help construction project managers to proactively identify construction mistakes, decide on corrective actions, and minimize cost and delays due to performance discrepancies [1].

Augmented Reality (AR) is an emerging technique that allows users to see real-world physical objects and their associated cyber-information overlaid on top of imagery of them. Mobile augmented reality is a variant of augmented reality that uses a mobile device's camera to capture real-world imagery and a mobile device's sensors to derive what cyber-information should be visible in the camera imagery, as shown in Figure 1.1. A key challenge of mobile augmented reality is that it relies on precisely localizing a user in order to determine what is visible in their camera view. The localization must be performed in the field without constraining the individual's whereabouts to a specially equipped area such as custom augmented reality "caves" with pre-deployed external infrastructure for location tracking. In other words, mobile augmented reality must work regardless of users' location and environment, and deliver relevant cyber-information precisely and quickly.

(a) (b)

Figure 1.1 An example of mobile augmented reality applications, (a) Facility management application, (b) Tourism application

Several key characteristics directly determine the reliability and utility of mobile augmented reality approaches: 1) user localization, which determines the users' viewpoint and derives what real-world physical objects that are in current scene, in order to interpret users' surrounding contexts and deliver relevant cyber-information to users, 2) the speed of determining which cyber-information is associated with physical objects in order to deliver/visualize the cyber-information in the correct position, 3) the usability of methods for creating 3D cyber-information and associating it with relevant real-world physical objects using mobile devices, 4) the robustness of the system and ability to work with dynamically changing environments, and 5) the scalability of the cyber-physical information association system, both in terms of physical scale, such as size of objects, and in terms of cyber-information scale, such as total number of cyber-information entities associated with physical objects. The purpose of this study is to address key research gaps in each of these areas that are not filled by current state-of-the-art augmented reality

research approaches.

Over the past decade, many research projects related to mobile augmented reality have focused on the first key component, i.e., accurate user localization, to realize mobile augmented reality on various types of mobile devices [2-6]. Prior localization approaches have primarily used Global Positioning Systems (GPS), Wireless Local Area Networks (WLAN), or Indoor GPS for positioning the user within the physical world [7-10]. The main drawback of these Radio Frequency (RF) based location tracking technologies is their high degree of dependency on pre-installed infrastructure, such as GPS satellites or wireless transceivers, and susceptibility to noise in commodity mobile device hardware [11], which makes their applications either highly inaccurate or impractical to use in many cases. Some research has focused on developing infrastructure-independent location tracking approaches [12, 13]. These approaches are typically based on inertial measurements and make use of highly accurate accelerometers and gyroscopes which are attached to users. However, these sensor-based approaches suffer from accumulated drift errors which grow with the distance traveled by the users.

Accordingly, the vast majority of prior work on mobile augmented reality either requires external sensors or very high computing resources to achieve a high-level of localization accuracy, and thus do not work well with commodity smartphones. In addition, very little research has examined the scalability issues of mobile augmented reality and fast cyber-physical information association. Despite the recent advances in mobile devices,

commodity smartphones still have limited processing power, inaccurate GPS sensors, and noisy accelerometers or gyroscopes.

Given the recent popularity and rising availability of smartphones in United States, however, robust mobile augmented reality systems that operate on commodity smartphone platforms should be developed to expand the number of context-aware applications. This study seeks to develop new approaches, algorithmic techniques, and hybrid mobile/cloud computing architectures that 1) support augmented reality on commodity smartphones, 2) can rapidly associate cyber-information with arbitrary real-world 3D objects, 3) provide millimeter-accuracy information association in near real-time without requiring external sensors or environmental constraints, 4) are extremely robust and resistant to environmental changes, such as users are moving from outdoor to indoor where GPS or other RF signals are typically denied and cannot be used for localization, and 5) can dynamically scale the augmented reality services from room-level to city-level scale.

A key differentiator of this research is its use of image-based localization from smartphone camera sensors and ability to localize users with respect to arbitrary marker-less 3D objects. The proposed mobile augmented reality approach, called as Hybrid 4-Dimensional Augmented Reality (HD^4AR) [14-18], provides reliable identification of the location and orientation of the user based on photographs taken by existing and already available commodity smartphones. The HD^4AR not only provides the location and orientation of the user, but also provides high-precision visualization of semantically-rich 3D cyber-

information over real-world imagery in an augmented reality (AR) format. Rather than using imprecise mobile GPS and/or wireless sensors, as in existing mobile AR approaches, the HD^{4}AR allows users to take pictures using smartphones for accurate localization in 3D and high-precision augmentation.

The remainder of this dissertation is organized as follows: After demonstrating open research problems in mobile augmented reality through a motivating example in Chapter 2, Chapter 3 presents prior research on mobile augmented reality and research gaps in prior work. In Chapter 4, technical details of the HD^{4}AR with empirical validation are discussed. The method for creating 3D cyber-information with a single 2D image is then illustrated in Chapter 5. In Chapters 6-7, new solution approaches for faster image-based localization/augmentation in large scale of usage, such as street-level mobile augmented reality, are presented. Specifically, Chapter 6 discusses a cached approach for the HD^{4}AR and Chapter 7 discusses the method for combining and/or clustering 3D point cloud models used in the HD^{4}AR. Finally, the dissertation concludes by summarizing contributions and identifying possible future work in Chapter 8.

2 Key Challenges of Mobile Augmented Reality

In this chapter, a motivating example is provided to illustrate the challenges of associating cyber-information with real-world physical objects. Specifically, a construction progress monitoring process from the Architectural, Engineering, Construction and Facility Management (AEC/FM) domain is used as it typically requires millimeter-level association of cyber-information, such as 3D blueprints of construction plans, with real-world construction building elements in challenging environments that are continually changing. After the motivating example is presented, open research problems on cyber-physical information association system, i.e., mobile augmented reality system, are presented and discussed in the context of the example.

2.1 Motivating Scenario

As a motivating example, a scenario where a field engineer is concerned about the construction progress and quality of a concrete foundation wall will be discussed. With the current best practices on construction sites, as shown in Figure 2.1, the field engineer would return to a construction trailer or office and open 2D construction drawings (at best a 3D Building Information Model (BIM)), project specifications, and the schedule to find out when the construction of this element is expected to be finished and what is the required quality of the outcome. Once the drawings and/or 3D building model are opened, the field engineer must navigate the model to determine which, of possibly hundreds or thousands of

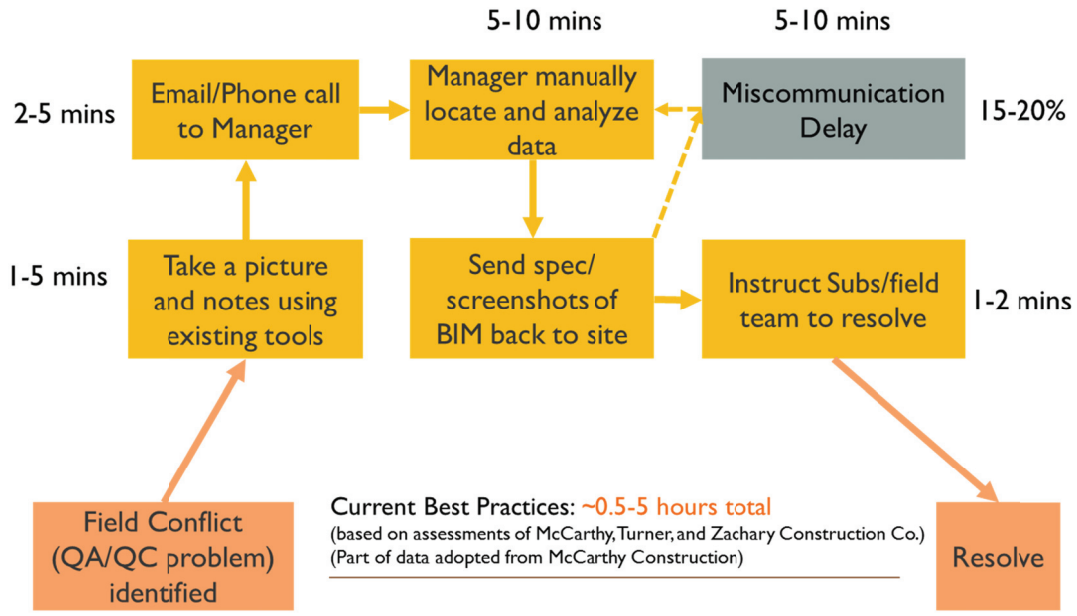

Figure 2.1 The current best practices for construction progress monitoring

walls, to figure out where the foundation wall of concern is. Moreover, once the information is obtained, the field engineer may need to return to the construction site to compare the information that was retrieved to the actual construction status of the real foundation wall. Because there is no easy way to directly query the cyber-information for the wall, the field engineer may not notice a discrepancy and will not be able to decide on a corrective action to minimize the impact of the discrepancy.

Instead, it would be beneficial if the field engineer can use the foundation wall itself to query for the needed building plan information directly from the site using a picture of the foundation wall as the basis for the query. This is exactly the type of real-world process where mobile augmented reality can be used to improve the speed and accuracy of decision

making. With mobile augmented reality, the picture from field engineer provides all information that is needed to localize the user with respect to their environment, and thus reduce the information available down to what is relevant to the current scene. Given the close proximity of construction elements, the location and orientation of the picture needs to be accurately estimated and relevant cyber-information should be precisely visualized and overlaid on top of each associated construction elements.

2.2 Open Research Problems in Mobile Augmented Reality

2.2.1 Problem 1: Need for Accurate User Localization

To deliver relevant building plan information to a field engineer, as described in the motivating example, the mobile augmented reality system first has to precisely identify his/her location to determine which construction elements are in the current viewpoint and how their associated 3D schematic and specifications should be visualized on top of photograph. More specifically, 3D localization, which identifies a user's position and orientation simultaneously, is required to deliver relevant information even with significant changes in the user's viewpoint, as shown in Figure 2.2. This 3D localization is often called 6-DOF (degrees-of-freedom) localization – three degrees from 3D rotational angles and three degrees from 3D translation distances. The accuracy of 6-DOF localization directly impacts the reliability of mobile augmented reality. In the context of the motivating example discussed in Section 2.1, the 6-DOF localization must be accurate to within 3-50 millimeters in order to correctly visualize a 3D schematic on top of a foundation wall. The

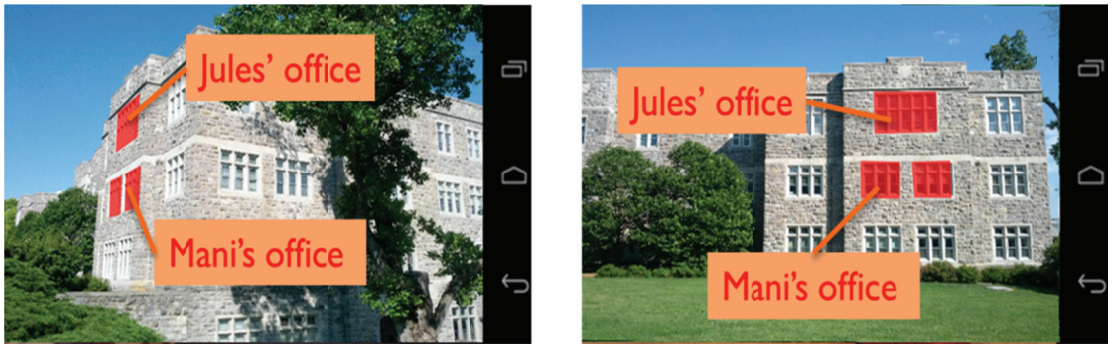

Figure 2.2 Needs for accurate user localization: cyber-information should be appeared precisely at significantly different viewpoint

current best practice for creating and visualizing 3D building plans on top of physical construction elements is to use high-end laser scanners, which typically provide single point position accuracy of approximately 12 *mm* and element recognition accuracy of 50 *mm* [1]. However, these approaches do not support mobility and on-site localization and their cost is in the $100,000 range. As a consequence, most research projects related to mobile augmented reality are primarily focused on accurate user localization with inexpensive mobile devices. Although the required level of accuracy depends on the target application, most mobile augmented reality systems have shown meter-level localization errors, which make them difficult to use in many practical scenarios, such as monitoring the manufacture of electronic circuit boards or monitoring construction progress. In addition, most prior research leverages specially manufactured sensors and/or devices due to the imprecision of commodity mobile device GPS sensors and noisy accelerometers.

2.2.2 Problem 2: Need for 3D Cyber-physical Content Authoring

Once the field engineer's location and his/her viewpoint are identified, a mobile augmented reality system then has to search for the associated 3D schematics or plan information that should appear in the current view. Figure 2.3 shows an example of 3D cyber-information associated with real-world physical objects. Since the delivery of relevant cyber-information to end-users is the ultimate goal of mobile augmented reality, the amount of available cyber-information determines the usefulness of the system. If there is no proper information to be delivered, localization results will be useless in the aspect of mobile augmented reality. Therefore, the capability of creating such 3D cyber-information and associating it with real-world physical objects is one of the key components in developing mobile augmented reality systems. For example, the system should provide a way of making notes on construction elements and associating them with real-world physical objects so that other engineers can see the notes overlaid on top of corresponding elements

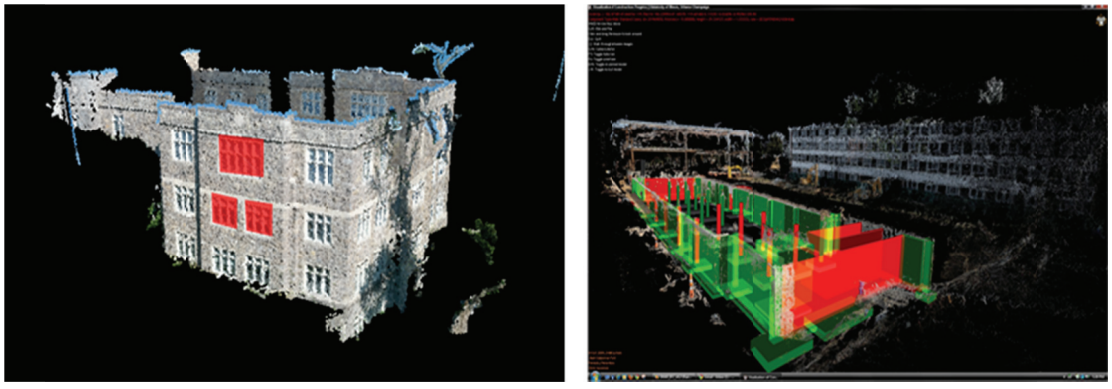

Figure 2.3 3D cyber-information associated with real-world physical models

through mobile augmented reality. However, the question of how to conveniently and accurately register even simple 3D content using a mobile device and 2D interface is still an open problem [19].

2.2.3 Problem 3: Need for Scalable Mobile Augmented Reality System

There are typically 5,000-30,000 building elements with their related specifications on a construction site and the physical scale of jobsites varies from tens of meters to hundreds of meters. Considering this variability, in terms of number of cyber-physical items and physical scales of target scene, it is difficult to design and implement general and near real-time mobile augmented reality system, especially with commodity smartphones which have limited resource of computing powers. Prior research has primarily tested their mobile augmented reality approaches on limited real-world use-cases, such as few office objects in the room. Up-to-date, there is no research that has analyzed scalability with respect to both the total number of cyber-information items and the physical scales of objects in the physical environment. Techniques are needed that can accurately operate at multiple different physical scales, such as on a remote control, indoor office room, large outdoor building, or entire outdoor street scene. In addition, the system should provide consistent and high-precision localization during the operation at dynamically changing scales and with large numbers of cyber-information items.

3 Related Work and Research Gaps

This chapter discusses the current state of knowledge and research gaps for each research problem outlined in Chapter 2, i.e., “Problem 1: Need for accurate user localization”, “Problem 2: Need for 3D Content Authoring and Association”, and “Problem 3: Need for Scalable Mobile Augmented Reality System”.

3.1 Research Gap 1: Fine-grained 3D Localization with Mobile Devices

3.1.1 Overview

Based on the techniques used for estimating a location and pose of the user’s mobile device, prior work on user localization can be roughly categorized into: 1) sensor-based localization which tracks the position using GPS and/or inertial, geomagnetic sensors attached to users, 2) marker-based localization which identifies the mobile device’s camera position and orientation by leveraging pre-defined optical markers and image processing techniques, 3) visual simultaneous localization and mapping (visual SLAM) which utilizes parallel threads for simultaneously tracking and mapping visual features from images, and 4) model-based localization which uses pre-constructed 3D models of the physical world as priori information to identify relative location and orientation of mobile devices. Table 3.1 summarizes and evaluates each category of prior research and presents qualitative assessment on localization accuracy and computational time. The desired values are based on the motivating scenario discussed in Chapter 2, i.e., real-world construction progress

Table 3.1 Qualitative comparison of localization techniques for mobile augmented reality systems

Metrics	**Sensor-based**	**Marker-based**	**Visual SLAM**	**Model-based**	**Desired**
Localization Accuracy	1.5 – 35 *m* [(a)]	0.5 – 2 *mm* [(b)]	0.5 – 20 *mm* [(c)]	0.5 – 20 *mm* [(c)]	Under 20 *mm*
Localization Speed	100 – 200 *msec*	20 – 140 *msec*	20 – 40 *msec*	5 – 240 sec	Under 3 *sec*
External Infrastructure	GPS satellite, RF transmitters	Optical markers	Not needed	Not needed	Not needed
Resistant to drifts and error accumulation	×	✓	×	✓	✓
Scale well to large scene	×	×	×	✓	✓
Supports mobility	✓	✓	✓	×	✓

[(a)] GPS Covered area; [(b)] Markers within 3*m* distance; [(c)] Objects within 10*m* distance.

monitoring scenario. For image-based localizations, such as marker-based, visual SLAM, and model-based localizations, the localization accuracy is typically computed in image pixel unit, i.e., projecting 3D objects or optical markers into mobile device's image sensor using recovered camera location and orientation and computing the pixel distance between projected points and corresponding image points where subjects actually appeared on the image, as shown in Figure 3.1. The measured image pixel errors, i.e. mean re-projection errors, can be converted to real-world distance metric, such as centimeters or millimeters, by using camera focal length, the dimension of camera image sensor, and the size of images. The details of image-based localizations will be further discussed in following subsection and the details of image pixel error conversion to real-world distance metric will be discussed in Section 4.4.

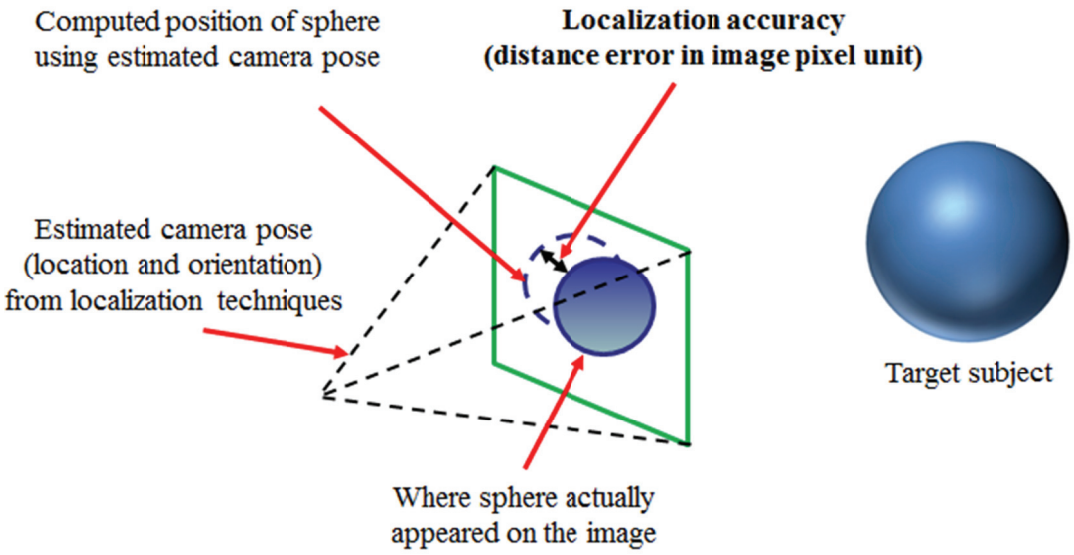

Figure 3.1 The definition of localization accuracy for image-based localizations

3.1.2 Gaps in Existing Research

The majority of prior work on user localization has relied on positioning systems, such as GPS or WLAN sensors [7, 9], or combined it with inertial measurers such as gyroscope sensors [12, 13]. Exploiting GPS sensors works well in outdoor environments but does not support indoor environments, and is unreliable in dense urban environments where a clear line of sight to the GPS satellite is unavailable. In addition, the use of GPS and inertial sensors in commodity smartphones introduces significant challenges due to the limited accuracy of GPS receiver and the noise presented in sensor data [11]. For example, the noise in geomagnetic heading values can cause jitter in onscreen information presentation. The indoor environment also imposes various challenges on location discovery due to dense multipath effects and building material dependent propagation effects. There are

many potential technologies and techniques that are suggested to offer the same functionality as a GPS indoors, such as WLAN, Ultra-Wide Band (UWB) and Indoor GPS. By tagging users with appropriate receivers/tags and deploying a number of nodes (e.g., access points, receivers, transmitters, etc.) at fixed positions indoors, the location of tagged users can be tracked by triangulation [12, 20]. However, the accuracy of using network infrastructure for 6-DOF localization is still questionable and their reliance on pre-installed infrastructures causing challenges in scalability.

In the meantime, several research groups have proposed marker-based mobile augmented reality to remove the dependency on mobile sensors or pre-installed network infrastructures [19-24]. These works track users' position and orientation using image processing techniques, i.e., matching the image captured by users' mobile devices to special, pre-defined 2D patterns (markers). Although marker-based localization has shown to work well in both indoor and outdoor environments and does not require additional sensors, yet the visual markers need to be attached to every real-world physical object of interest. Tagging hundreds to thousands of objects with 2D markers in the case of large-scale environments, such as street scenes, or construction site, is impractical and does not scale well to handle various distances to objects.

The advent of computer vision methods over the past decade has led to new research on the application of image-based localization methods for marker-less mobile augmented reality systems. Due to the independency on pre-installed infrastructure, inertial measurers, and/or

optimal markers, vision-based localization methods have gained significant attention in the computer vision community, as well as in the augmented reality community [1, 19, 25-36]. A group of these works have focused on visual Simultaneous Localization and Mapping (SLAM) [26, 29, 31], which simultaneously constructs a sparse 3D map from visual features and localizes a device using generated map, with parallel threads of tracking and mapping (PTAM) [28] method. However, visual SLAM methods mostly focus on small-scale environment, such as indoor office room, and suffer from inconsistent loop closure problem when the scale becomes larger, such as outdoor buildings on the street. In addition, in the context of augmented reality, the visual SLAM methods are difficult to associate arbitrary 3D cyber-information with physical objects as the 3D coordinates of the map are varying from the devices and their initial locations of calibration. As a consequence, the visual SLAM methods require either an offline-learned 3D model or manual association of 3D cyber-information, whenever users initiate the SLAM method with different devices. Another drawback of visual SLAM methods is that the performance of localization depends on the used devices. All the computations need to be done on the board of the devices, and thus, the localization speed relies on the computing power of mobile devices. The dependency on used mobile devices makes visual SLAM methods difficult to structure the general large-scale mobile augmented reality system, typically in form of server-client architecture, which allows people to collaboratively add or query cyber-information.

Finally, another group of computer vision based works has shown that a set of overlapping images can be used to extract very accurate 3D geometry of stationary subjects, such as

buildings under construction, in form of 3D point cloud model. After extracting the 3D point cloud of the subjects through the Structure-from-Motion (SfM) algorithm that estimates the 3D position of the visual features through image feature extraction, pair-wise matching, initial triangulation, and the Bundle Adjustment [32] optimization process, the 3D point cloud model can be used as a prior knowledge to compute 2D-to-3D correspondences for precisely localizing mobile camera imagery [33-36]. Using a 3D point cloud for user localization, i.e., model-based localization, permits mobile augmented reality systems to accurately estimate the 3D position and 3D orientation of the new photograph purely based on the image [14-18], and therefore, it does not have any hardware constraints on mobile devices, such as stereo cameras, GPS sensors, or motion tracking sensors. Furthermore, recent advances in SfM [37-39] enable the easy creation of large scale 3D point clouds from an unordered set of images and extend model-based localization methods to large scene such as street-level or city-level scale.

Although this body of computer vision research has shown the potential and high-accuracy of model-based reasoning, most of the recent model-based localization methods assume that those point clouds are already available at the beginning of the localization process. The 3D point cloud generation process, also called as 3D reconstruction, is often separated from the localization process and the 3D reconstruction is done in an offline preparation step. Despite the scalability of recent approaches in SfM, however, collecting image data and processing them to prepare a 3D point cloud still takes considerable amount of time. The Bundler [39], a widely-used SfM software package, takes from hours to a day to

generate a 3D point cloud even with small numbers of input images. This time-consuming preparation of 3D point cloud prevents using model-based localization in mobile augmented reality, especially when users want to model a daily changing scene such as construction site. Furthermore, the low speed of model-based localization (typically 5 – 240 *sec*) and the lack of on spot localization methods make their applications difficult to use in mobile augmented reality. One of the objectives of the proposed research is to overcome these challenges in model-based localization methods by optimizing both 3D reconstruction and localization processes, and make it available on mobile devices to provide near real-time mobile augmented reality.

3.2 Research Gap 2: 3D Cyber-physical Content Authoring from a 2D interface

3.2.1 Overview

Another important capability in mobile augmented reality is being able to author and associate cyber-content with the real-world physical objects around the user. Prior work has assumed that this content is already available and focused on mobile augmented reality systems with fast and accurate user localization. Creating and associating cyber-information with physical objects on-the-fly, however, is challenging due to the complexity of spatially associating cyber-information with the geometry of arbitrary real-world objects, such as engine parts or windows on the building, in a 3D space and using a small 2D mobile device interface.

Existing work on 3D content authoring can be roughly categorized into: 1) 3D drawing methods which use 3D design tools to create 3D contents, 2) gesture recognition based methods, which track the motions of the users' fingers or other tools in order to draw 3D contents into the virtual 3D space. All of these methods require specific devices to support 3D content authoring and manual association with real-world physical objects, and thus, do not work well with commodity smartphones. Table 3.2 summarizes and evaluates 3D cyber-physical content authoring methods in each category and presents metrics for qualitative comparison.

3.2.2 Gaps in Existing Research

Despite the great efforts to facilitate onsite activities through mobile augmented reality, most of research has mainly focused on retrieving existing cyber-information and displaying them over imagery captured by mobile devices in form of augmented reality

Table 3.2 Qualitative comparison of 3D content authoring techniques for mobile augmented reality systems

Metrics	**3D drawings**	**Gesture recognition**	**Desired**
External 3D framework	CAD	Not needed	Not needed
Device type	Personal Computer	Gloves, pens	Commodity smartphones
Automatic association with real-world objects	×	×	✓
Supports mobility	×	✓	✓

overlays without proposing methods that can easily and quickly create cyber-information and associate it with real-world physical objects.

A number of 3D drawing methods for content authoring have been discussed by several works for mobile augmented reality systems [1, 3, 12, 24, 25]. However, all of these works used existing commercial 3D drawing tools to create 3D cyber-information and manually geo-tagged or aligned cyber-information to real-world physical objects. The main problem with this approach is that it requires specific 3D design frameworks (e.g., Computer Aided Design (CAD) tool) and devices (e.g., mouse, pen, etc.), which are not available on commodity mobile devices.

More recently, some research have focused on intuitive methods for 3D content authoring, such as gesture recognition based methods [40, 41]. These methods track the movements of users' fingers or pen, create virtual objects corresponding to those movements, and visualize them on top of the camera view. Although they provide more straightforward methods than 3D drawing based methods, the user interface is still complicated and difficult to draw 3D virtual objects accurately. In addition, they also require special devices, such as gloves or sensor-attached pens, and do not provide automatic association of created cyber-information with real-world physical objects. One of the objectives of the proposed research is to overcome these limitations and provide a practical method for 3D cyber-physical content authoring on a 2D mobile device interface with no external hardware.

3.3 Research Gap 3: Near Real-time Cyber-Physical Information Association at Dynamically Varying Environmental Scales

3.3.1 Overview

Since model-based localization methods provide sufficient accuracy for high-precision cyber-physical information association scenarios, such as identifying the buttons on a car dashboard, overlaying construction information on walls, etc., this study focuses on model-based localization techniques for high-precision mobile augmented reality systems. In addition, model-based localization techniques are only approaches that do not require any external infrastructures, such as GPS satellites, wireless network sensors, or fiducial/optical markers, as described in Section 3.1. As a consequence, existing work on model-based localization methods has been analyzed for performance comparison at different environmental scales, such as room-level, or street-level. Table 3.3 summarizes and evaluates existing model-based localization techniques and presents metrics for qualitative comparison.

3.3.2 Gaps in Existing Research

Lim et al. [35] and Sattler et al. [36] proposed near real-time model-based localization methods. However, their test cases consist of only a single 3D point cloud model at room-level scale and their approaches were not true mobile augmented reality as they were unable to provide cyber-information delivery/visualization functionality and the mobility.

Table 3.3 Qualitative comparison for scalability of mobile augmented reality systems

Metrics	**Model-based**	**Desired**
Model scale	room-street	object-street
Model preparation time	3 – 24 *hr*	0.1 – 1 *hr*
Number of 3D physical models in the system	Single	Multiple (Hundreds of models)
Number of cyber-information items in the system	$0 – 10^3$	$10^0 – 10^4$
Localization/ Augmentation Speed	5 – 240 *sec*	Under 3 *sec*
Supports mobility	×	✓

Applications of model-based localization methods in augmented reality systems can be found in [1, 25]. These systems were designed for context-aware AEC/FM applications to enhance construction progress monitoring processes. The 3D point cloud model is generated from pre-collected photographs of a construction site and the system uses the extracted model at street-level scale to localize users. Although their systems precisely determine the users' location and deliver relevant construction project information to end-users, yet it could not conduct user localization in the field for on-site decision making purposes. With their systems, field personnel have to take photographs and bring them back to the office to process each photograph. Even after field personnel bringing photographs back to the office, localizing a single photograph to see the cyber-information overlaid on top of imagery takes tens of seconds with a high-end personal computer at the office. Considering the applications and the current limits from these works, a new approach,

which takes at most 1-3 seconds regardless of operating scales and provides mobility with commodity smartphones, should be developed.

In addition, all aforementioned works on model-based localization were based on the 3D point clouds generated by the SfM framework in the Bundler package. To produce a single 3D point cloud, the Bundler package typically takes from hours to days depending on the number of input images, due to exhaustive computations in pair-wise feature matching and non-linear multi-dimensional Bundle Adjustment optimization on a single-thread CPU. This considerable amount of time for 3D point cloud preparation also prevents developing general mobile augmented reality systems using model-based localization for dynamically varying environmental scales. The details of model preparation, i.e., 3D reconstruction, will be further discussed in Section 4.2.

4 Hybrid 4-Dimensional Augmented Reality (HD^{4}AR)

4.1 Overview of Solution Approach to Research Gaps 1 and 3

To fill the "Research Gap 1: Fine-grained 6-DOF Localization with Mobile Devices", and "Research Gap 3: Near Real-time Cyber-physical Information Association at Dynamically Varying Environmental Scales", a new type of mobile augmented reality, Hybrid 4-Dimensional Augmented Reality (HD^{4}AR), is proposed and developed. The HD^{4}AR uses a model-based localization approach, which takes advantage of a pre-constructed 3D point cloud of target scene to identify a mobile device's relative location and orientation. Since the 3D point cloud generated from a set of overlapping photographs represents an accurate 3D geometry of real-world physical objects, it is often called as 3D physical model. Consequently, the HD^{4}AR requires a 3D reconstruction process that rapidly and robustly generates a 3D physical model from pre-collected photographs.

As discussed in Section 3.1, using a 3D physical model for localization permits the system to estimate the complete pose (6-DOF) of the camera, and therefore can support high-accuracy augmented reality applications, such as construction progress monitoring where millimeter-level precision is needed. Due to time-consuming preparation of 3D point cloud [1, 25, 37-39], however, using a 3D physical model for localization is often considered as an impractical solution for mobile augmented reality. To overcome this challenge, a new parallelized 3D reconstruction process, which combines different image feature descriptors, operates across cores in a multi-core CPU and GPU for fast operations, and thus is suitable

for mobile augmented reality, is designed and developed. The algorithmic details and enhancements of the new fast 3D reconstruction process will be presented in Section 4.2.

Once the 3D physical model is available, a user can take a new photo at a random location and his/her location and orientation are determined by comparing the new image to the generated 3D physical model. Specifically, the system attempts to estimate extrinsic camera parameters, i.e., a rotation matrix and a translation vector of the camera, to find the relative position of the user's camera (mobile device). After recovering a complete pose of the user's camera, the system can decide what cyber-information should appear in the user's photograph. However, existing model-based localization methods take tens of seconds even with a high-end personal computer to localize a single image, which is not suitable for mobile augmented reality with commodity mobile devices. Therefore, a new model-based 6-DOF localization method using a direct 2D-to-3D matching algorithm, which takes at most few seconds to localize a photograph, is devised and developed. In addition, the HD^{4}AR uses the client-server architecture to further increase the localization speed. The smartphone as the client uploads new photographs to the server for localization and the major image processing load is located on the server. The details of a new model-based localization method will be discussed in Section 4.3. Figures 4.1 and 4.2 summarize the overall procedures of the HD^{4}AR, from initial 3D reconstruction to localization/ augmentation process.

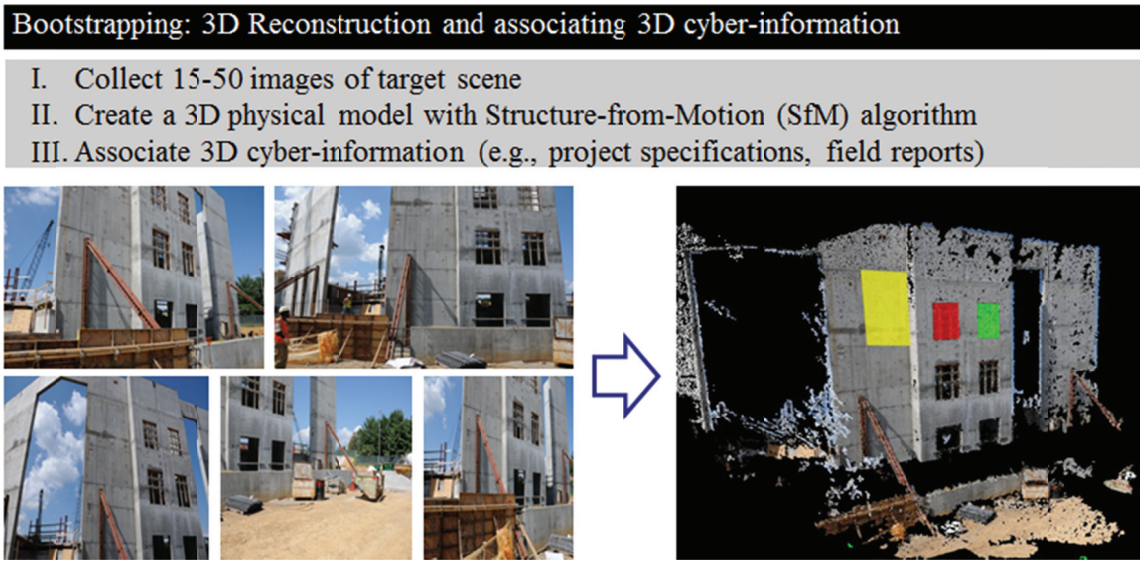

Figure 4.1 Initial 3D Reconstruction: Bootstrapping of HD^{4}AR

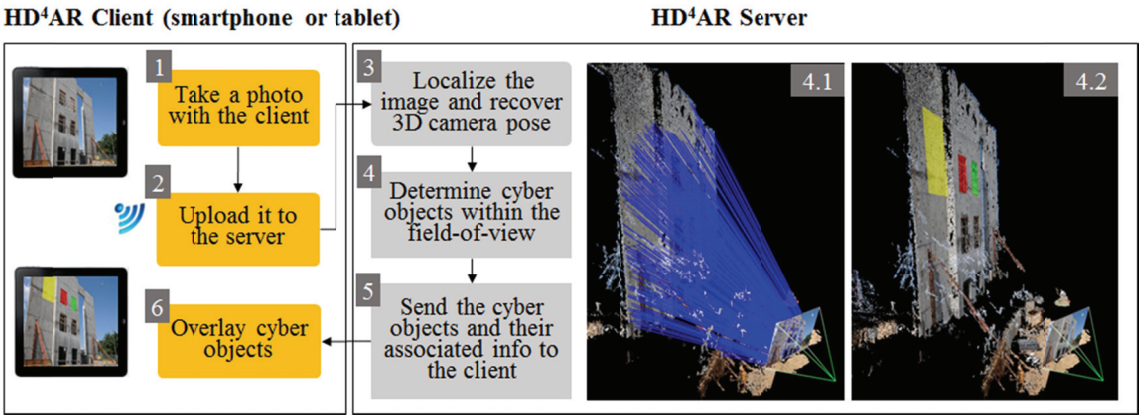

Figure 4.2 Model-based 6-DOF localization and augmentation of HD^{4}AR

4.2 New Parallelized Structure-from-Motion for 3D Physical Model Generation

As described in Section 4.1, an initial 3D point cloud must be created to serve as a reference model for model-based localization and/or mobile augmented reality. In addition,

this 3D physical model must be generated quickly for fast initialization of the system. Generating this 3D physical model requires a collection of overlapping base images of the target scene and processing these images using the SfM algorithm that estimates the 3D positions of 2D image feature points.

To accelerate the speed of SfM-based 3D reconstruction, computer vision researchers have proposed several methods separately from mobile augmented reality applications and none of these works are feasible for mobile augmented reality using 3D physical models. First, the Bundler package has been developed by Snavely et al. [39]. Snavely et al. have created the first structured pipeline for 3D point cloud modeling from an unordered set of large-scale internet photo collections. The Bundler uses the SIFT (Scale Invariant Feature Transformation) descriptor [42] for feature extraction, which has good invariance properties but requires multiple layers of computation for each spatial scale, and thus is time consuming. In addition, the pair-wise image matching in the Bundler is performed on a single-thread CPU, and therefore the processing time grows exponentially with the size of image set. More recently, a cloud computing scheme has been introduced to accelerate the entire SfM procedure [37]. A cloud computing has achieved a remarkable performance gain on very large-scale 3D reconstruction by distributing tasks over several hundreds of cores. However, using several hundreds of cores is often not feasible and the system is still based on CPU-based SIFT descriptor. Another approach uses both GPU-based SIFT and an image clustering scheme on a cloudless system [38]. The proposed system, however, limits the number of feature points per image due to memory bandwidth of the GPU and its purpose

is estimating the pose of base cameras to recover the surface of the scene rather than creating an accurate 3D point cloud for user localization or augmented reality.

To further speed up 3D reconstruction task and enable fast initialization of mobile augmented reality system, a new parallelized SfM framework, which supports new types of feature descriptors to replace the time-consuming SIFT descriptor, is developed and used in the HD^{4}AR. Compared to vector-based real-number descriptors, such as SIFT or SURF (Speeded Up Robust Features) [43], the HD^{4}AR takes advantage of binary feature descriptors, which consist of a binary bit-string rather than a vector of real-numbers, to reduce memory consumption and computational complexity of image processing in both 3D reconstruction and localization. The advantages of using binary descriptors are that 1) it requires much less memory than real-number descriptors and 2) it can use the Hamming distance for descriptor matching, which is faster than the Euclidian distance comparison. However, binary descriptors are typically considered as a trade-off, providing less robustness against image rotation or scaling. While some research have compared the robustness of binary descriptors against 2D image rotation and scaling, no research has argued the impact of binary descriptors on 3D reconstruction and compared different feature descriptors using a single unified SfM-based 3D reconstruction framework. Through the extensive experiments, we realize that recently proposed binary descriptors, such as BRISK (Binary Robust Invariant Scalable Keypoint) [44] or FREAK (Fast REtinA Keypoint) [45], have a strong potential for accurate 3D reconstruction. As a consequence,

CPU-based SIFT, GPU-based SURF, CPU-based BRISK and CPU-based FREAK are comprehensively analyzed and compared within the HD^{4}AR.

A new filtering approach is also developed for accurate 3D reconstruction and the structure of 3D physical model is optimized for further application, such as fast model-based localization and/or mobile augmented reality. In addition, an entire 3D reconstruction process exploits hardware/software parallelism including parallelized nearest neighbor searching to scale the performance of 3D reconstruction. The proposed parallelized SfM framework follows some of the original algorithmic steps in [39], but significantly alters others in order to vastly accelerate the process, improve robustness, and improve accuracy.

As aforementioned, the key modifications that make the most substantial impact on performance are: 1) the combination of different feature detectors and descriptors to optimize the 3D reconstruction performance, 2) new filtering approach for reducing noise in the 3D point clouds and improving localization accuracy, 3) memory-efficient point cloud structure for mobile augmented reality and 4) a parallelized multicore CPU and GPU hardware implementation for faster processing. Figure 4.3 illustrates the overview of the HD^{4}AR 3D reconstruction process, consisting of four algorithmic stages. The details of each algorithmic stage are further discussed in the following subsections.

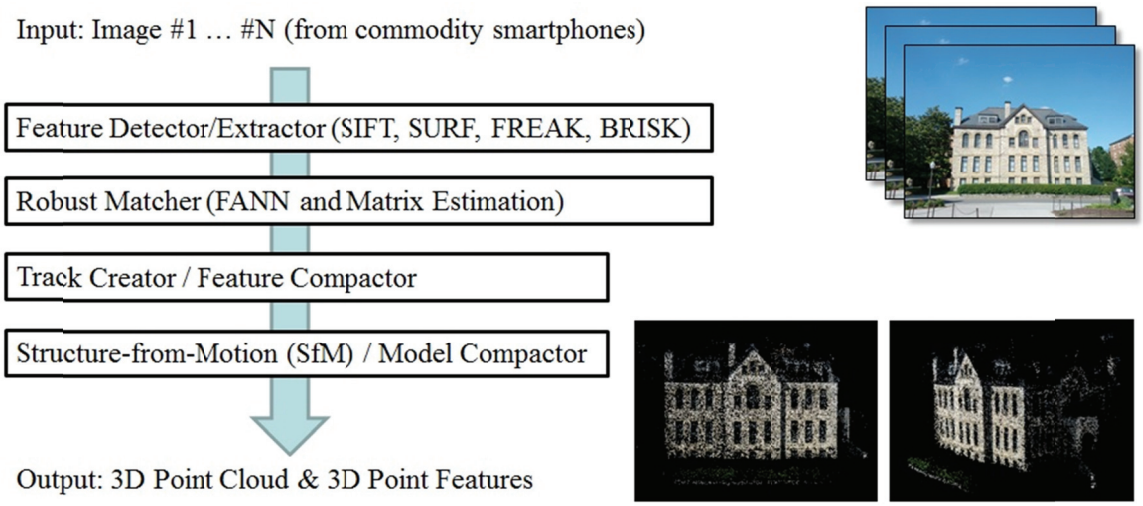

Figure 4.3 New parallelized SfM process for mobile augmented reality

4.2.1 Feature Detection/Extraction Stage

The first stage of the 3D reconstruction is the *Feature Detection/Extraction* process which extracts image keypoints and feature descriptors for each base image. Figure 4.4 shows the overall structure of the *Feature Detection/Extraction* stage. To find a set of image keypoints, a feature detection algorithm is first run on each input image. The CPU-based SIFT and GPU-based SURF are implemented and used in the *Detector* module. Both SIFT and SURF are invariant to image scaling and rotation and thus appropriate for 3D reconstruction from unordered photographs. However, the SIFT and SURF algorithms use slightly different ways of detecting feature points. The SIFT builds a set of image pyramids and filters each layer with Difference of Gaussians (DoG) [42]. On the other hand, the SURF creates a stack without downsampling for higher levels in the pyramid and it filters the stack using a

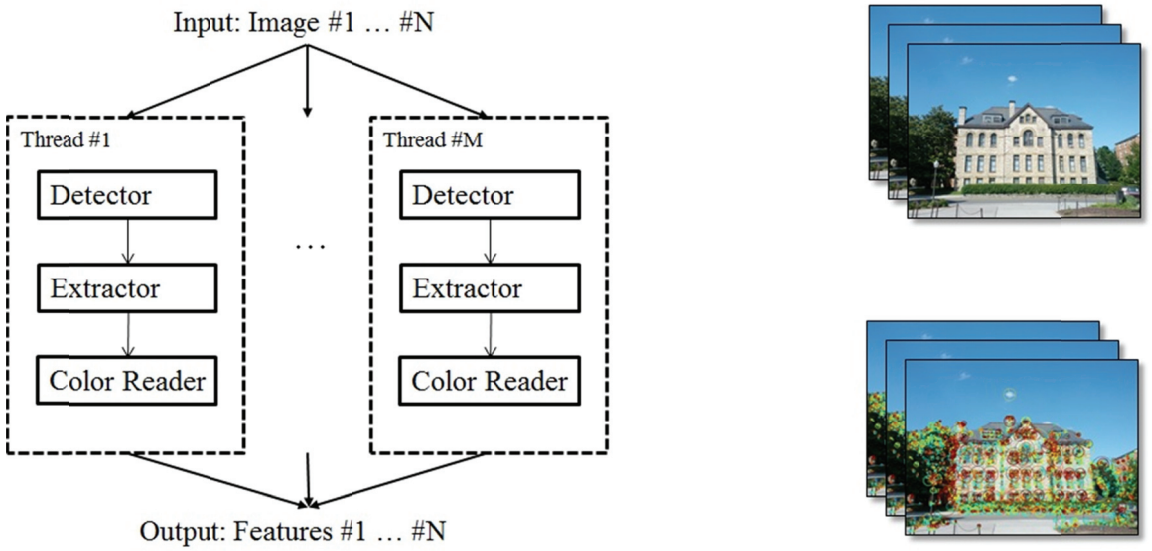

Figure 4.4 Overall structure of *Feature Detection/Extraction* stage

box filter approximation of second-order Gaussian partial derivatives to speed up the processing time [43].

Next, the *Extractor* module extracts feature descriptors at the detected image keypoints. These extracted feature descriptors will be used as the basis for pair-wise image matching. The CPU-based SIFT, GPU-based SURF, CPU-based FREAK and CPU-based BRISK are implemented and used in this module. In contrast to SIFT and SURF, the FREAK uses retinal sampling patterns to compare image intensities and produces a cascade of binary strings [44]. The BRISK also assembles a bit-string descriptor from intensity comparisons retrieved by dedicated sampling of each keypoint neighborhood [45]. These resulting binary descriptors consume much less disk space compared to vector-based real-number descriptors, such as SIFT and SURF, and use the Hamming distance instead of Euclidian

distance for descriptor matching. After extracting feature descriptors, the pixel color information of detected keypoints is read by the *Color Reader* module. The color information will be used later to assign colors to each 3D point for visualization purpose. Then, all outputs are stored as binary files for faster Input/Output (I/O) tasks.

To investigate how feature detector and feature descriptor affect the performance and quality of 3D reconstruction, we have tested four different detector-descriptor combinations in our experiments, i.e., SIFT-SIFT, SURF-SURF, SURF-FREAK, and SURF-BRISK. To simplify the name of these combinations, we refer to them as SIFT, SURF, FREAK, and BRISK, respectively. Figure 4.5 shows invariant properties of each combination against 2D image rotation and scaling. From this simple test result, we can infer that all these combinations will work well for 3D reconstruction. The detailed experimental results of 3D reconstruction are presented and fully discussed in Section 4.4.

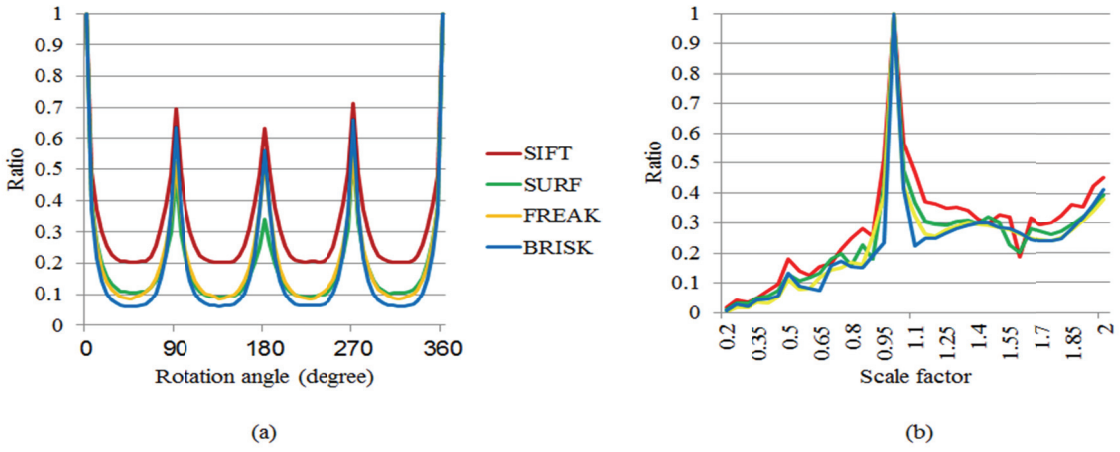

Figure 4.5 Results of descriptor invariance test on a real-world imagery: (a) rotational invariance test, (b) scaling invariance test

4.2.2 Robust Matching Stage

The next step is finding correspondences between all image pairs (i.e., pair-wise matching) using extracted feature descriptors. For binary feature descriptors (FREAK and BRISK), the *FANN Matcher* module first creates hierarchical clustering *k-d* trees of each image descriptors and runs the Fast Approximate Nearest Neighbors (FANN) searching algorithm [46] to rapidly find the two nearest neighbors of each descriptor in the image. For vector-based real number descriptors (SIFT and SURF), the *FANN Matcher* module runs randomized *k-d* tree searching algorithm with four parallel trees to improve the search speed [47]. With all recovered nearest neighbor results, the *FANN Matcher* module finally performs the distance ratio-test [42] with threshold value 0.5 to remove suspicious matches. In addition, if more than one feature descriptor matches the same feature in the opposite image, it removes all of matches for that image pair.

After the distance ratio-test, the *F-matrix* module robustly estimates a fundamental matrix and further removes outliers for every image pair using the RANSAC (RANdom SAmple Consensus) algorithm with the eight-point algorithm [48]. This filtering process removes false matches using an epipolar geometry constraint given by the estimated fundamental matrix. In other words, the maximum allowed distance from a keypoint to an epipolar line is σ_F pixels, beyond which the point is considered as an outlier. This outlier constraint can be expressed as:

$$|| \mathbf{x_i}^T \mathbf{F_{ij}} \mathbf{x_j} || > \sigma_F = \max(\max(w_i, h_i), \max(w_j, h_j)) \times 0.006 \quad (4.1)$$

where $\mathbf{x}_i = [u_i, v_i, 1]^T$ and $\mathbf{x}_j = [u_j, v_j, 1]^T$ are homogenous coordinates of the matched keypoints in image *i* and *j*, respectively, $\mathbf{F}_{ij}$ is the estimated Fundamental matrix from RANSAC iterations, and (w_i, h_i) and (w_j, h_j) are the dimension of image *i* and *j*, respectively. If the number of final inliers is less than 16, all of the matches are removed for that image pair. Otherwise, the fundamental matrix returned by RANSAC is further refined by running the Levenberg-Marquardt algorithm, minimizing the distance to the epipolar line for all the inliers.

Upon receiving the inliers from the *F-matrix* module, the *H-matrix* module finds a homography matrix using the RANSAC with normalized Direct Linear Transform [48] for every image pair. The outlier constraint is in the form of

$$\| \mathbf{x_i} - \mathbf{H_{ij}}\mathbf{x_j} \| > \sigma_H = \max(\max(w_i, h_i), \max(w_j, h_j)) \times 0.004 \quad (4.2)$$

where $\mathbf{x}_i = [u_i, v_i, 1]$ and $\mathbf{x}_j = [u_j, v_j, 1]$ are homogenous coordinates of the inliers after fitting to fundamental matrix, and $\mathbf{H_{ij}}$ is the estimated homography matrix from RANSAC iterations, and (w_i, h_i) and (w_j, h_j) are the dimension of image *i* and *j*, respectively. Then, the percentage of number of inliers with homography matrix, *H-score*, is calculated and recorded. The *H-score* will be used in final *Structure-from-Motion* stage and image-based cyber-content authoring method to select the proper image sets.

Since the pair-wise image matching is the most performance bottleneck in 3D

reconstruction, each image pair is processed on different threads with lock-free parallelization of the FANN searching to shorten the overall processing time. Figure 4.6 shows the overall structure of the *Robust Matching* stage. Due to the FANN searching and multi-threading of the tasks, the performance of pair-wise matching is significantly improved compared to an existing SfM package, e.g., the Bundler package.

4.2.3 Track Creation/Feature Compaction Stage

The *Track Creation/Feature Compaction* stage first creates tracks from matching results, where a track is a connected set of matching keypoints across multiple images. Figure 4.7 illustrates the overall procedures of this stage. Through extensive experiments, we found

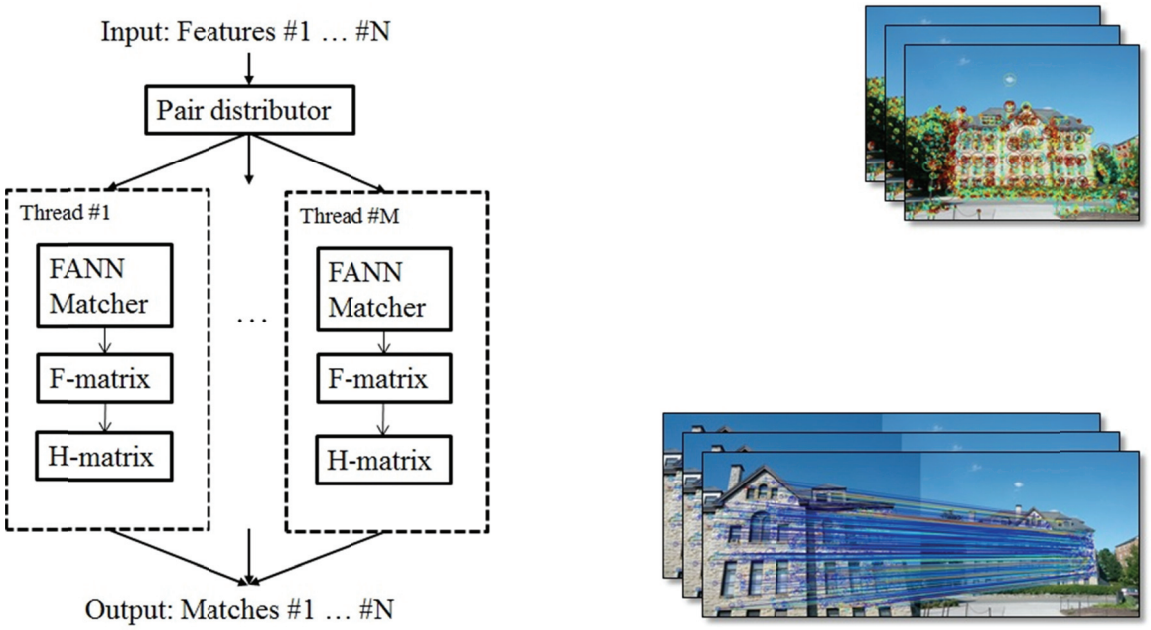

Figure 4.6 Overall structure of *Robust Matching* stage

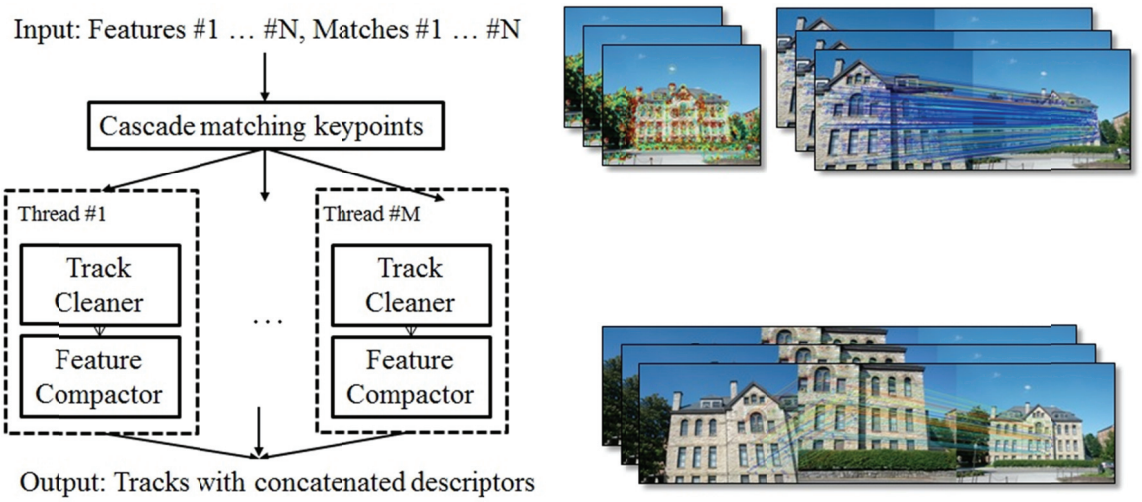

Figure 4.7 Overall structure of *Track Creation/Feature Compaction* stage

that some false matches can still survive in the matching stage even after robust tests, such as distance ratio-test and fitting to the fundamental matrix, were performed. This situation is likely to happen when the target scene has repeated patterns, such as multiple similar windows in the building. If these surviving false matches are organized into tracks, the SfM procedure may generate a very noisy 3D point cloud model.

Therefore, we have designed and included a track ratio-test in the track creation stage to remove false matches from each track by comparing all the matching distances of the keypoints inside the track. If one of the matching keypoints connected to a track has very high distance than others, that keypoint is erased from the track. In other words, the *Cleaner* module removes a keypoint from the track if

$$d_m / d_k < \sigma_{TR} \tag{4.3}$$

where d_m is the minimum matching distance among all keypoints in the track and d_k is the matching distance of each keypoint in the track. We call this procedure as a track ratio-test and the σ_{TR} is typically set to 0.3. In addition to the track ratio-test that removes the inconsistent keypoints for each track, the *Cleaner* module also removes inconsistent tracks by observing the length of each track. If the length of a track is less than σ_{TL}, which means that the track is seen by only σ_{TL} cameras, that track will not be considered in 3D reconstruction. The σ_{TL} can be set to 3 or 4 for very accurate 3D modeling if the input photographs were taken with specific purpose and have numerous overlapping images of target scene. However, the σ_{TL} is typically set to 2 since we target an unordered set of photographs taken at random locations.

Finally, the *Feature Compactor* module extracts and merges the feature descriptors of keypoints that are remaining in the set of consistent tracks. This process significantly reduces the disk space consumption as well as the speed of I/O task in the next stage.

4.2.4 Structure-from-Motion/Model Compaction Stage

The final stage of the HD^{4}AR 3D reconstruction is the *Structure-from-Motion (SfM)*/*Model Compaction* stage that estimates a set of camera parameters, such as focal length, radial distortion coefficient, rotation matrix, and translation vector, for each base image and a 3D location for each track. Similar to the Bundler, this stage uses an incremental approach, recovering a few cameras at a time. Once the 3D point cloud is reconstructed, the *Structure-from-Motion (SfM)/Model Compaction* stage also extracts and imposes a representative

feature descriptor for each 3D point, making 3D point clouds ready for direct 2D-to-3D matching used in model-based localization. Figure 4.8 shows the overall structure of the stage and Figure 4.9 shows an example of 3D point clouds generated by the proposed framework using real-world construction element and static building photos.

The *SfM* stage first starts with an initial image pair to recover camera parameters using Nistér's five-point algorithm [49], and triangulates their feature points using polynomial method [50]. As discussed in [39], this initial pair should have a large number of matched

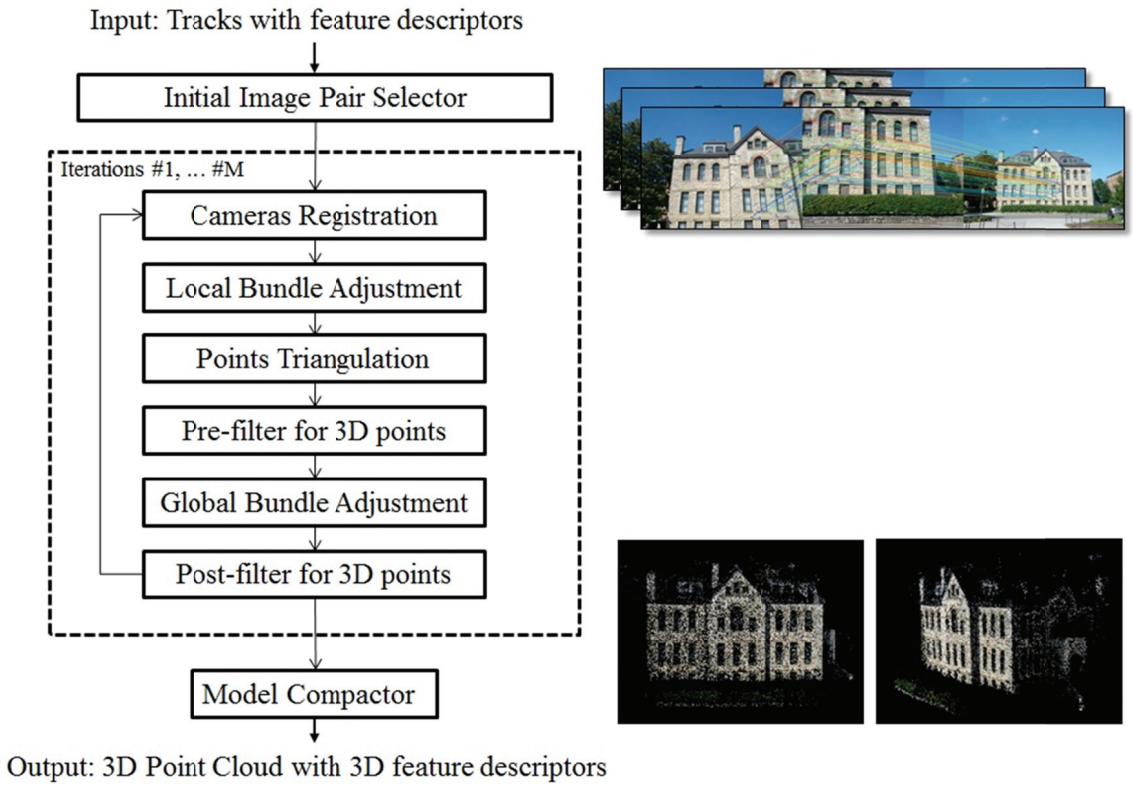

Figure 4.8 Overall structure of *Structure-from-Motion/Model Compaction* stage

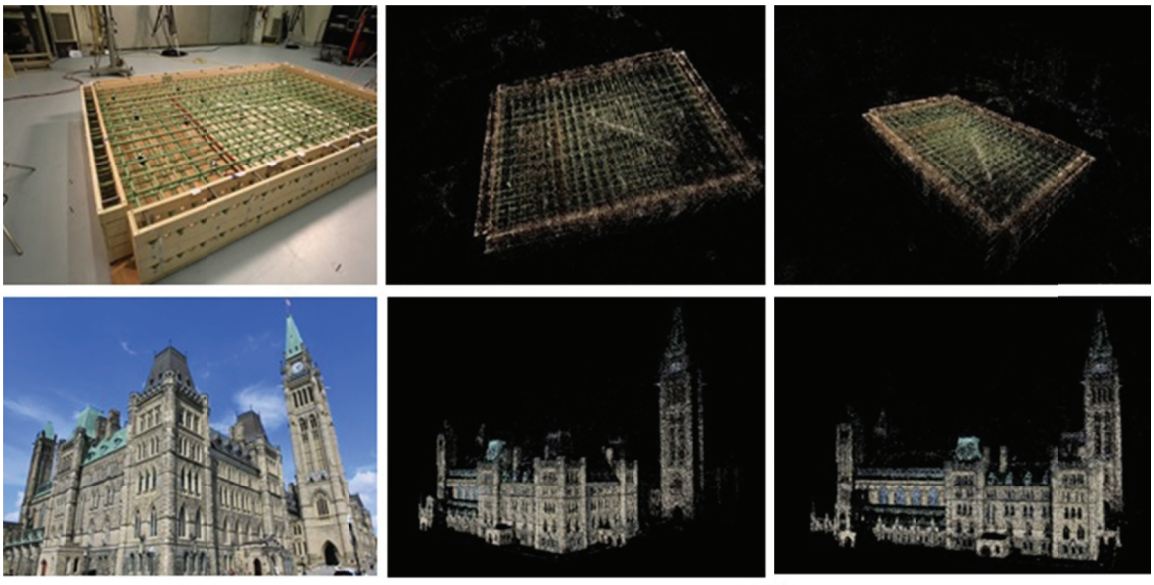

Figure 4.9 3D physical models from the HD[4]AR 3D reconstruction: (a) initial base images, (b) 3D point clouds – resulting 3D point clouds well-represent the target construction element and building.

feature points, but also have a long separation distance between the cameras to avoid getting stuck in local minimum during the optimization process. To fulfill this requirement, the *SfM* module selects an initial image pair which has the lowest *H-Score* among all possible pairs of images. However, our experiments have shown that the *H-score* should be greater than 0.25 and the number of matches between selected pair should be greater than 200 to generate the most accurate 3D point cloud. Therefore, these conditions are also taken account into initial image pair selection. After calibrating the camera parameters and triangulating feature points of initial image pair, the Bundle Adjustment optimization [32] is run to minimize the overall mean re-projection error, i.e., the difference between predicted 2D positions of the feature points in the photographs given their triangulated 3D positions

and the locations of where the feature points are actually extracted in the images. To significantly enhance the speed of this optimization, we adopt a GPU-based Parallel Bundle Adjustment approach [51].

Then, the SfM algorithm goes through iterations to calibrate camera parameters of each additional input image using the already triangulated 3D points and matching information between the images. This calibration is done using PnP (Perspective n-Point) camera estimation method with RANSAC and Levenberg-Marquardt optimization [48]. If the algorithm successfully recovers camera parameters of an additional base image, it registers the new camera and runs the Local Bundle Adjustment, i.e., optimizing only the newly added cameras. This camera registration fails in the event that an additional input image does not have any matched feature points against the previously registered images. After Local Bundle Adjustment, the component triangulates the 3D points seen by the newly registered cameras and pre-filters 3D points which have high re-projection error. Through extensive experiments, we realized that this pre-filtering step is vital for accurate 3D modeling. Very little number of high-error 3D points can destroy an entire shape of 3D point cloud even with the Bundle Adjustment which tries to minimize overall mean re-projection error. The outlier threshold for this pre-filtering based on re-projection error is set to the same value used in the *F-matrix* module of the *Robust Matching* stage.

Finally, the Global Bundle Adjustment is run to optimize entire 3D points currently retrieved and all parameters of currently registered cameras. During this optimization,

however, it is possible that some 3D points still have a high re-projection error while other 3D points have a very small re-projection error, resulting in a small mean re-projection error. The ultimate purpose of the 3D reconstruction is user localization and/or mobile augmented reality, not the visual representation of target scene, it is very important to reduce such noise in the 3D point cloud by removing 3D points with a high re-projection error. To achieve this, the SfM algorithm in the HD^{4}AR uses a double-threshold scheme for the post-filtering stage. The first threshold is for controlling the target Mean Squared Error (MSE) of the Bundle Adjustment. This threshold value is set to be 0.25 pixel^2 so that the target average re-projection error of entire 3D point cloud is equal to 0.5 pixels. Another threshold, which called an absolute re-projection threshold, is for removing individual 3D points from a 3D point cloud. The absolute re-projection threshold is adaptively calculated based on the current distribution of re-projection errors of each base image. Nevertheless, the maximum value of this threshold is set to be 4.0 pixels so that no 3D points in the final 3D point cloud have a re-projection error greater than 4.0 pixels. After post-filtering stage, if the registered camera has the number of visible 3D points less than 16, that camera is removed from 3D reconstruction as it will not provide an accurate estimation of camera parameters due to small number of visible points. The entire SfM procedure including the Global Bundle Adjustment and post-filtering is iteratively executed until there are no more cameras to register. Due to the algorithmic enhancements and parallelization, the HD^{4}AR 3D reconstruction is up to 30 times faster than the Bundler package. In Section 4.4, experimental results of this new SfM algorithm are discussed in detail.

Once the 3D points and camera parameters of input images are successfully recovered, the *Model Compactor* module finally collects image feature descriptors for all triangulated tracks and creates a representative descriptor for each 3D point to enable direct 2D-to-3D matching. As described in [15, 16, 18, 36], a direct 2D-to-3D matching method have a considerable potential for fast and accurate model-based localization. We propose to use minimum-distance criteria, rather than averaging image descriptors proposed by Sattler et al. [36], as the HD^{4}AR should be able to handle binary descriptors, such as FREAK or BRISK. The process of generating 3D representative descriptors can be summarized as follows:

For each 3D point ($\mathbf{X}_n$) in the 3D point cloud model,

1) Find a list of base images ($\mathbf{I}_1$, …, $\mathbf{I}_k$) and their corresponding 2D image points ($\mathbf{x}_1$, ..., $\mathbf{x}_k$) that participated in triangulation of $\mathbf{X}_n$ during the 3D reconstruction.
2) Collect image feature descriptors ($\mathbf{d}_1$, …, $\mathbf{d}_k$) at discovered 2D image points ($\mathbf{x}_1$, …, $\mathbf{x}_k$), where each descriptor is typically a 64-dimensional (SURF, FREAK, BRISK) or 128-dimensional (SIFT, SURF) vector.
3) For each descriptor in ($\mathbf{d}_1$, …, $\mathbf{d}_k$), sum Hamming (FREAK, BRISK) or Euclidean (SIFT) distances to all other descriptors in the set.
4) Select the descriptor, which has the minimum summation value, as a representative descriptor of the 3D point ($\mathbf{X}_n$).

Due to this representative descriptors approach, the localization time will depend on the

number of 3D points in the point cloud, not on the number of input images used in 3D reconstruction, resulting faster localization compared to existing model-based localization methods. The details of new direct 2D-to-3D matching for model-based localization will be discussed in Section 4.3.

4.3 Model-based 6-DOF Localization/Augmentation Using Direct 2D-to-3D Matching

4.3.1 Hybrid Mobile/Cloud Architecture

Once the HD^{4}AR has the 3D physical model of the target scene, it can accurately localize and augment new photographs captured by a mobile device. Figures 4.2 and 4.10 summarize this process from a high-level perspective. As shown in Figure 4.10, the HD^{4}AR uses the client-server architecture – with the mobile devices as the client – to upload images taken from the mobile devices to the server for 3D reconstruction and user localization purposes. The entire system consists of the following components:

- *Client application*: the HD^{4}AR client application runs on Android or iOS devices. This application captures the images and uploads them to the server. It also has the capability of drawing cyber objects on top of a single image and attaching arbitrary documents as cyber objects.
- *Server – image-based 3D reconstruction*: this component generates a SfM-based 3D point cloud from initial base images and runs on a cloud computing platform.

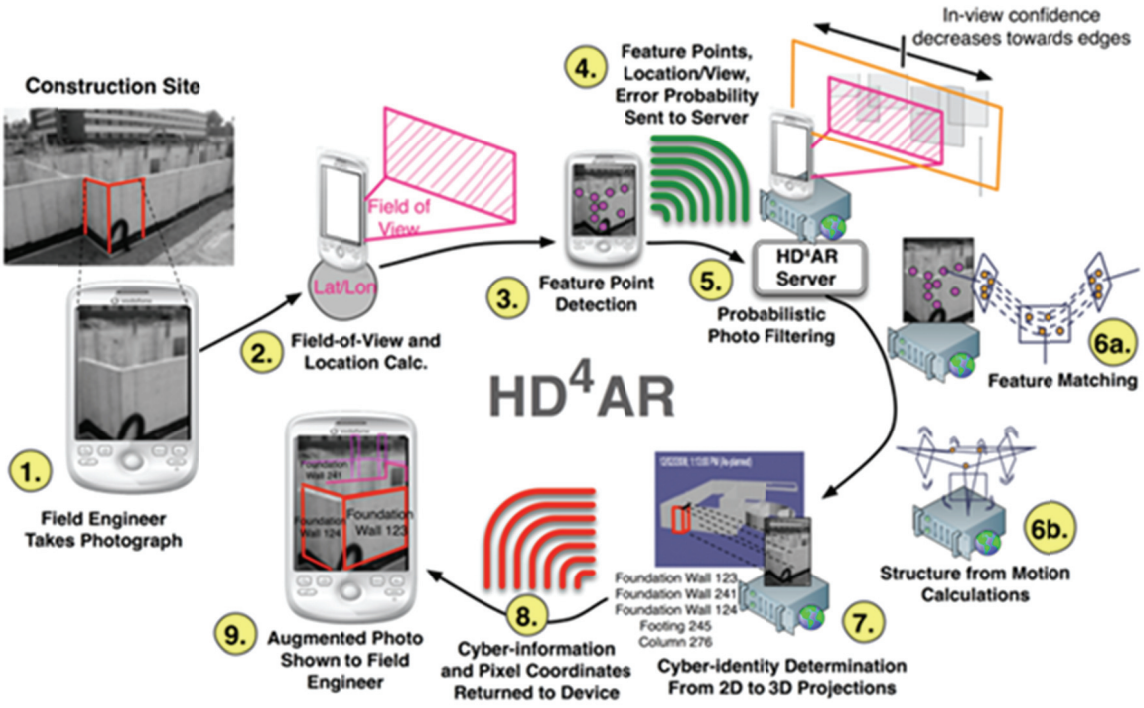

Figure 4.10 The client-server architecture of HD^{4}AR and the sequence of localization/augmentation

The initial base images can be uploaded to the server via the HD^{4}AR client app or a web-based interface.

- *Server – user localization*: this component takes a single image captured on a mobile device as input and derives a 3D position and orientation of the mobile device with respect to the 3D point cloud by solving a Direct Linear Transform (DLT) equation followed by a Levenberg-Marquardt optimization [48].
- *Server – back-projection*: this component takes the position of the 2D cyber objects in the photograph as input and computes the 3D position of cyber information using the underlying 3D point cloud model and calibrated camera parameters. The details of back-projection and content authoring method will be discussed in Chapter 5.

By developing a new direct 2D-to-3D matching algorithm, which will be presented in following subsection, and putting major image processing load on the server, the model-based localization and augmentation of the HD^{4}AR can be done in near real-time. Furthermore, due to the client-server architecture, the performance of localization does not depend on the computing power of the mobile devices, and thus, the system can easily support multiple types of commodity mobile devices if devices have a capability of camera imaging and network communication.

4.3.2 Direct 2D-to-3D Matching with 3D Physical Model

To localize a user and display surrounding cyber-information on top of the imagery, a user first takes a picture of the objects, which he/she wishes to query for information about, and uploads the photograph to the HD^{4}AR server. Upon receiving the photo from user's mobile device, the server runs feature detection and extraction on the received image, finds correspondences between the image and the underlying 3D physical model. Finding 2D-to-3D correspondences between the 2D feature points detected on the new image captured by a mobile device and the 3D points in the physical model can be accelerated using a direct 2D-to-3D matching algorithm. While existing works match feature descriptors of the image to an entire set of feature descriptors from all base images used in 3D reconstruction to find correspondences (2D-to-2D-to-3D matching), which incurs unnecessary descriptor comparisons, the HD^{4}AR only compares feature descriptors of the image to the representative descriptors of each 3D point in the 3D physical model, resulting in near real-time localization and augmentation. The representative descriptors of 3D point cloud model

(see Section 4.2.4) are cached in the server in form of a *k-d* tree structure and Fast Approximate Nearest Neighborhood (FANN) searching algorithm [46, 47] is used for rapidly finding correspondences between 2D image feature descriptors and 3D representative descriptors. In addition, the proposed method of direct 2D-to-3D matching and extracting 3D representative descriptors work well for both vector-based real-number descriptors and the binary descriptors. Due to this new direct 2D-to-3D matching, the localization time now depends on the size of the 3D physical model, i.e., the number of 3D points, not on the number of base images used in 3D reconstruction. In addition, this approach does not only create representative descriptors of 3D points, but also provides higher probability of finding 2D-to-3D correspondences as it selects the descriptor, which has the minimum distance across all base images, as a representative descriptor for each 3D point. As we will discuss in Section 4.4, the proposed direct 2D-to-3D matching approach speeds up the localization by a factor of 162 compared to the Bundler.

After discovering 2D-to-3D correspondences, the camera calibration algorithm is performed by solving Direct Linear Transformation (DLT) equation followed by a Levenberg-Marquardt optimization [48]. This model-based camera calibration results in 6-DOF (degrees-of-freedom) localization in 3D space and thus gives high localization accuracy despite possible variation in the position and orientation of the user within the reconstructed scene. If the server successfully estimates the camera pose information, it determines what cyber-information is within the camera's field of view and where the

information should appear. This decision is done by first projecting each vertex of 3D cyber-information onto the localized camera:

$$s\begin{bmatrix} x \\ y \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} [R \quad | \quad T] \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix} \tag{4.4}$$

where $[X, Y, Z, 1]^T$ is a 3D vertex point of cyber-information, [R|T] is an estimated 3x3 rotation matrix and an estimated 3x1 translation vector, $(f_x\ f_y)$ is a camera focal length expressed in pixel units, $(c_x\ c_y)$ is a principal point of the camera, and $[x, y, 1]^T$ is a resulting projected points in image pixels. Next, the simple visibility test is performed to determine whether the 3D cyber-information appears in current image or not.

$$V(x,y) = \begin{cases} 1, & 0 \le x \le w, 0 \le y \le h \\ 0, & Otherwise \end{cases} \tag{4.5}$$

where w is image width and h is image height. The visible cyber-information is then sent back to user's mobile device with positional information and semantics. Finally, the user's mobile device renders the returned visible cyber-information on the top of captured-image. As shown in Figure 4.11, the HD^{4}AR can precisely localize and augment photographs with various test cases and it implies that the HD^{4}AR remains stable under different viewpoint of the user's mobile device.

Figure 4.11 HD^{4}AR localization and augmentation: cyber-information is precisely overlaid on user's photograph despite the significant change of viewpoint.

4.4 Experimental Results and Validation

This section presents experimental results and the validation of the proposed mobile augmented reality system – HD^{4}AR. As described in Sections 4.1-4.3, the HD^{4}AR combines model-based localization with SfM-based 3D point cloud model, and therefore, two separate experiments, i.e., 3D reconstruction and model-based 6-DOF localization, were performed and validated. In order to assess improvements provided by the HD^{4}AR, each experimental result was compared to the result from the Bundler package, the most widely-used SfM package using incremental approach. The details of the data set specifications and validation metrics are discussed in the following subsections. After showing experimental results, the overall validation of the solution approach will be summarized.

4.4.1 3D Reconstruction

The 3D reconstruction experiments were conducted on a single Amazon EC2 instance server with 22.5 GB memory and two Intel Xeon X5570 processors running Ubuntu version 12.04. An NVIDIA Tesla M2050 graphic card was used for GPU computations. The image data sets used to create the 3D point clouds can be roughly categorized as: 1) outdoor: construction site or existing buildings on the street, and 2) indoor: car interior, kitchen, or office room. Table 4.1 presents the summary of data sets that cover different scales of target objects and scenes.

Table 4.1 Dataset specification for 3D reconstruction

Environment	Name	Scale description	Number of base images	Image resolution	Camera model
outdoor	patton	building	40	2592 × 1944	Samsung Galaxy Nexus
	knu	building	50	2592 × 1458	Samsung Galaxy Nexus
	parliament	landmark	52	4752 × 3168	Canon EOS 50D
	rtfr	construction site	113	3264 × 2448	Apple iPhone 4S
	cfta	construction site	80	2144 × 1424	Nikon D300S
	rh	construction site	155	2144 × 1424	Nikon D300
indoor	dashboard	car dashboard	27	2592 × 1944	Samsung Galaxy Nexus
	engine	car engine parts	32	3264 × 2448	Apple iPhone 4S
	kitchen	home kitchen	47	2048 × 1536	Samsung Galaxy Nexus
	ikea	office store	44	3264 × 2448	Apple iPhone 4S

An entire 3D reconstruction procedure of the HD^{4}AR was run on each data set to produce the 3D physical models. To demonstrate the performance gains of the HD^{4}AR resulting from track compression, double-threshold filtering, parallelized matching, etc., the following metrics were measured:

- *Number of registered images*: how many pre-collected photographs were calibrated. This metric measures the completeness of the 3D reconstruction process if the data set was properly collected. Higher numbers of calibrated cameras will increase the reliability of the positional information of 3D points triangulated during the 3D reconstruction.
- *Number of 3D points*: how many 3D points were successfully triangulated. Larger numbers of 3D points increase the probability of direct 2D-to-3D matching and 3D localization success for mobile augmented reality.
- *Mean re-projection error*: overall mean re-projection error is computed by projecting each 3D point into each calibrated camera of the base images in order to measure the positional error of generated 3D physical models. This metric measures the robustness and accuracy of the 3D physical model and affects the accuracy of 3D localization for mobile augmented reality.
- *Point cloud size*: how much disk space is consumed by a single 3D physical model. The point cloud size is a key concern if multiple physical models are cached in the server simultaneously.
- *Elapsed time*: how long does it take to generate a single 3D physical model. A

specific aim of our framework was reducing this time in order to rapidly enable mobile augmented reality using 3D point cloud models.

Tables 4.2-4.4 compare the overall results of 3D reconstruction on the outdoor building photographs, i.e., “patton”, “knu”, and “parliament” data sets. Although there are many factors that influenced the performance, such as the number of base images, the image sizes, and the texture of the target scenes, the HD^{4}AR 3D reconstruction was 304-2,875% faster than the Bundler for all building-scale outdoor data sets we studied. The performance gain was significant when binary descriptors, i.e., the FREAK and BRISK, were used. Specifically, the HD^{4}AR achieved 1,169-2,875% of performance gain with binary descriptors, and produced 3D physical models within 20 *min*. Even with same SIFT descriptor used in the Bundler package, the HD^{4}AR was up to 9.419 times faster.

Next, the HD^{4}AR significantly reduces the memory consumption of 3D physical models as it only records the representative descriptors of each 3D point, while the Bundler stores all feature descriptors from the entire set of base images. In addition, the Bundler uses the SIFT descriptor, which is 128-dimensional real-number vector, so it consumes a lot of disk space to store information related to 3D physical models for localization (called registration in the Bundler) and mobile augmented reality. Specifically, the HD^{4}AR achieved 1,860-2,759% of memory gain with binary descriptors. Memory consumption is important when multiple mobile clients perform online localization simultaneously with different 3D physical models. Large file sizes prevent from pre-loading multiple models into memory

Table 4.2 Performance of 3D reconstruction for “patton” data set

Package	**Bundler**	**HD^{4}AR**			
Descriptor	**SIFT**	**SIFT**	**SURF**	**FREAK**	**BRISK**
Number of registered images	40 / 40	40 / 40	40 / 40	40 / 40	40 / 40
Number of 3D points	129,693	147,798	72,000	47,163	46,318
Mean re-projection error	0.661 pixels	0.578 pixels	0.596 pixels	0.502 pixels	0.498 pixels
Point cloud size (memory gain)	446.90 MB (1×)	331.00 MB (1.35×)	72.80 MB (6.14×)	16.30 MB (27.42×)	16.20 MB (27.59×)
Elapsed time (performance gain)	8,571 *sec* (1×)	2,824.424 *sec* (3.035×)	923.932 *sec* (9.277×)	300.358 *sec* (28.536×)	298.095 *sec* (28.753×)

Table 4.3 Performance comparison of 3D reconstruction for “knu” data set

Package	**Bundler**	**HD^{4}AR**			
Descriptor	**SIFT**	**SIFT**	**SURF**	**FREAK**	**BRISK**
Number of registered images	50 / 50	49 / 50	50 / 50	49 / 50	49 / 50
Number of 3D points	37,356	51,730	40,858	32,827	33,122
Mean re-projection error	0.681 pixels	0.504 pixels	0.673 pixels	0.595 pixels	0.552 pixels
Point cloud size (memory gain)	223.16 MB (1×)	104.00 MB (2.15×)	41.38 MB (5.39×)	12.02 MB (18.57×)	11.97 MB (18.64×)
Elapsed time (performance gain)	4,424 *sec* (1×)	469.687 *sec* (9.419×)	314.944 *sec* (14.047×)	321.040 *sec* (13.78×)	378.303 *sec* (11.694×)

Table 4.4 Performance comparison of 3D reconstruction for "parliament" data set

Package	**Bundler**[(a)]	**HD^{4}AR**			
Descriptor	**SIFT**	**SIFT**	**SURF**	**FREAK**	**BRISK**
Number of registered images	-	52 / 52	52 / 52	52 / 52	52 / 52
Number of 3D points	-	431,559	273,166	223,886	234,343
Mean re-projection error	-	0.649 pixels	0.674 pixels	0.604 pixels	0.606 pixels
Point cloud size (memory gain)	-	0.99 GB (-)	328.15 MB (-)	91.77 MB (-)	96.01 MB (-)
Elapsed time (performance gain)	-	10,800 *sec* (-)	1,396.002 *sec* (-)	1,332.656 *sec* (-)	1,279.646 *sec* (-)

[(a)] The Bundler failed to create 3D point cloud due to image size and out of memory problem.

and reduce server-side localization speed due to increased disk I/O and memory swapping. In our experience, the file I/O for reading 3D physical model for localization takes about 6 *sec* when the 3D point cloud size excesses 300 MB, and it is about 70% of the entire model-based localization process if the server does not cache the point cloud in the memory.

Finally, the mean re-projection errors show that the HD^{4}AR generated more accurate 3D point clouds for the building-scale outdoor data sets. The HD^{4}AR achieved mean re-projection errors less than 0.673 pixels and less than the results from the Bundler for all cases. The mean re-projection error represents how accurate the resulting 3D point cloud and the calibrated camera parameters are, as the re-projection error is calculated by projecting each 3D point into each calibrated camera of the base images and computing the

distance to the position of original image feature point. The experimental results illustrate that the generated 3D point clouds with the HD^{4}AR have only 1-pixel mean re-projection error and well-represent the target scenes.

One interesting result is that the Bundler failed to create a 3D physical model for the "parliament" data set. As shown in Table 4.1, the "parliament" images were taken by a high-end DSLR camera, and therefore, the images are very high-resolution with large file sizes. During the 3D reconstruction with these high resolution images, the Bundler package caused the out of memory problem and could not process the data set. However, as shown in Table 4.4, the HD^{4}AR well-handled the "parliament" data set and successfully produced the dense large-scale 3D physical models. Except the SIFT descriptor, the HD^{4}AR only took about 20 *min* to generate hundreds of thousands 3D points. Figure 4.12 shows the generated 3D physical models from all building-scale outdoor data sets using the BRISK descriptor.

While binary descriptors achieved a huge gain on both reconstruction speed and memory consumption on the outdoor data sets, they produced little less dense 3D point clouds. The outdoor images typically have a plenty of textures and therefore, the invariance properties of feature descriptors shown in Figure 4.5 affect the number of true matches between photographs taken at random location and orientation. A key question is whether or not the reduction in point cloud density impacts mobile client localization. Based on visual analysis of the point clouds presented in Figure 4.12, we believe that the reduced density of the 3D

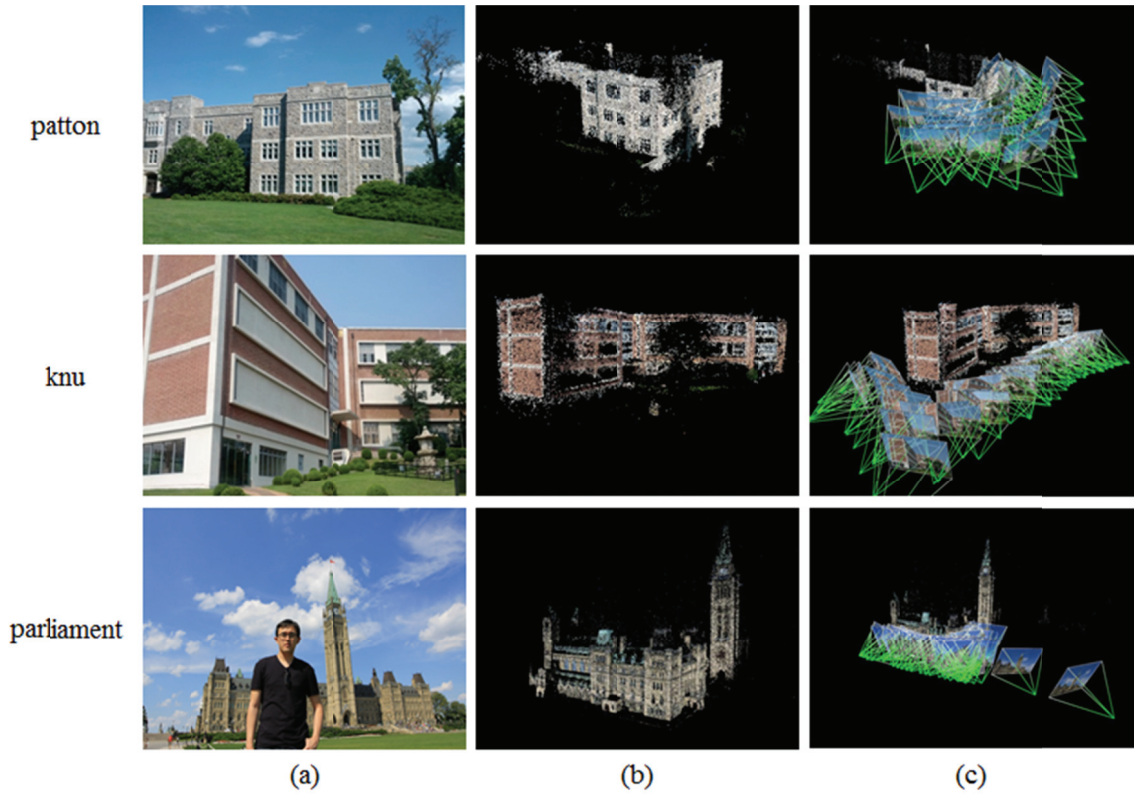

Figure 4.12 3D reconstruction results for building-scale outdoor data sets with BRISK descriptor: (a) initial base images, (b) 3D point clouds from the HD^{4}AR, and (c) 3D point clouds with estimated camera position of input base images

point clouds would not affect model-based 6-DOF localization since all 3D point clouds well-represent the target scenes. Rather, the smaller number of 3D points accelerates the direct 2D-to-3D matching by focusing on the most significant feature points and therefore improves localization speed. The performance of localization will be further discussed in following subsection.

Tables 4.5-4.7 compare the overall results of 3D reconstruction on the outdoor construction jobsite photographs, i.e., “rtfr”, “cfta”, and “rh” data sets. These data sets were collected on

the jobsites during real-world construction activities. Again, the HD^{4}AR outperformed the Bundler and was 594-1,639% faster for outdoor construction data sets. In addition, the HD^{4}AR achieved the memory gain up to 1,740% and all generated 3D physical models have mean re-projection error smaller than 1.379 pixels. Figure 4.13 shows the generated 3D physical models from all outdoor construction data sets using the BRISK descriptor.

As demonstrated in Tables 4.5-4.7 and Figure 4.13, we can conclude that the HD^{4}AR successfully generates 3D physical models for street-scale construction jobsites, even with the binary feature descriptors.

Table 4.5 Performance comparison of 3D reconstruction for "rtfr" data set

Package	**Bundler**	**HD^{4}AR**			
Descriptor	**SIFT**	**SIFT**	**SURF**	**FREAK**	**BRISK**
Number of registered images	113 / 113	112 / 113	113 / 113	112 / 113	113 / 113
Number of 3D points	48,493	40,526	81,197	81,140	81,909
Mean re-projection error	1.375 pixels	1.254 pixels	1.086 pixels	1.379 pixels	1.356 pixels
Point cloud size (memory gain)	263.27 MB (1×)	89.97 MB (2.93×)	79.31 MB (3.32×)	35.57 MB (7.40×)	36.80 MB (7.15×)
Elapsed time (performance gain)	14,989 *sec* (1×)	1,535.326 *sec* (9.763×)	2473.289 *sec* (6.060×)	1740.730 *sec* (8.611×)	1832.693 *sec* (8.179×)

Table 4.6 Performance comparison of 3D reconstruction for “cfta” data set

Package	**Bundler**	**HD^{4}AR**			
Descriptor	**SIFT**	**SIFT**	**SURF**	**FREAK**	**BRISK**
Number of registered images	80 / 80	80 / 80	80 / 80	80 / 80	80 / 80
Number of 3D points	29,164	10,680	10,627	12,266	15,042
Mean re-projection error	0.698 pixels	0.594 pixels	0.709 pixels	0.580 pixels	0.615 pixels
Point cloud size (memory gain)	133.60 MB (1×)	41.11 MB (3.25×)	13.80 MB (9.68×)	7.82 MB (17.08×)	10.23 MB (13.06×)
Elapsed time (performance gain)	5,086 *sec* (1×)	600.155 *sec* (8.474×)	855.769 *sec* (5.943×)	698.884 *sec* (7.277×)	447.473 *sec* (11.366×)

Table 4.7 Performance comparison of 3D reconstruction for “rh” data set

Package	**Bundler**	**HD^{4}AR**			
Descriptor	**SIFT**	**SIFT**	**SURF**	**FREAK**	**BRISK**
Number of registered images	155 / 155	155 / 155	155 / 155	149 / 155	151 / 155
Number of 3D points	59,533	27,247	36,854	31,738	41,097
Mean re-projection error	0.818 pixels	0.603 pixels	0.703 pixels	0.567 pixels	0.600 pixels
Point cloud size (memory gain)	247.08 MB (1×)	60.00 MB (4.12×)	38.80 MB (6.37×)	14.20 MB (17.40×)	18.10 MB (13.65×)
Elapsed time (performance gain)	16,070 *sec* (1×)	980.450 *sec* (16.390×)	2475.513 *sec* (6.494×)	1329.698 *sec* (12.085×)	1371.612 *sec* (11.716×)

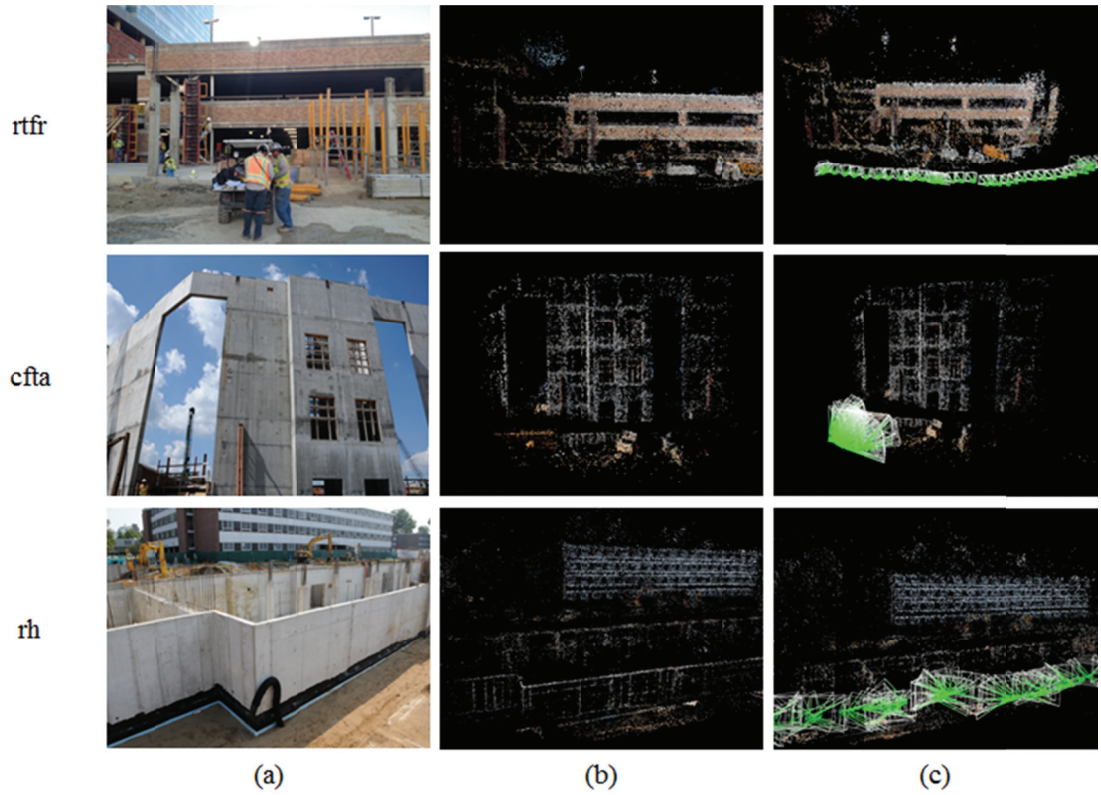

Figure 4.13 3D reconstruction results for street-scale construction jobsites with BRISK descriptor: (a) initial base images, (b) 3D point clouds from the HD^{4}AR, and (c) 3D point clouds with estimated camera position of input base images

To assess the capability of indoor 3D reconstruction, various images were collected and processed with the HD^{4}AR. Tables 4.8-4.11 compare the overall results of 3D reconstruction on the indoor data sets, i.e., “dashboard”, “engine”, “kitchen”, and “ikea”. For the indoor data sets, the HD^{4}AR was 661-1,558% faster than the Bundler and achieved the memory gain up to 3,242%. Regardless of used feature descriptors, all generated 3D physical models have mean re-projection errors within the range between 0.644 and 1.284 pixels, while the Bundler has the errors up to 2.308 pixels. In addition, all 3D point clouds were generated within 3 *min* for indoor data sets we studied.

Table 4.8 Performance comparison of 3D reconstruction for “dashboard” data set

Package	**Bundler**	**HD^{4}AR**			
Descriptor	**SIFT**	**SIFT**	**SURF**	**FREAK**	**BRISK**
Number of registered images	27 / 27	27 / 27	27 / 27	27 / 27	27 / 27
Number of 3D points	5,210	5,806	9,179	7,962	5,962
Mean re-projection error	0.881 pixels	0.677 pixels	0.967 pixels	0.767 pixels	0.755 pixels
Point cloud size (memory gain)	34.64 MB (1×)	12.10 MB (2.86×)	8.83 MB (3.92×)	2.80 MB (12.37×)	2.17 MB (15.96×)
Elapsed time (performance gain)	736 *sec* (1×)	93.031 *sec* (7.911×)	111.330 *sec* (6.611×)	104.675 *sec* (7.031×)	60.373 *sec* (12.191×)

Table 4.9 Performance comparison of 3D reconstruction for “engine” data set

Package	**Bundler**	**HD^{4}AR**			
Descriptor	**SIFT**	**SIFT**	**SURF**	**FREAK**	**BRISK**
Number of registered images	10 / 32	31 / 32	21 / 32	12 / 32	10 / 32
Number of 3D points	6,708	35,292	28,051	10,381	9,653
Mean re-projection error	2.166 pixels	0.692 pixels	0.756 pixels	0.650 pixels	0.644 pixels
Point cloud size (memory gain)	101.80 MB (1×)	63.74 MB (1.60×)	25.70 MB (3.96×)	3.44 MB (29.59×)	3.14 MB (32.42×)
Elapsed time (performance gain)	2,007 *sec* (1×)	225.727 *sec* (8.891×)	196.240 *sec* (10.227×)	167.355 *sec* (11.992×)	179.663 *sec* (11.171×)

Table 4.10 Performance comparison of 3D reconstruction for “kitchen” data set

Package	**Bundler**	**HD^{4}AR**			
Descriptor	**SIFT**	**SIFT**	**SURF**	**FREAK**	**BRISK**
Number of registered images	47 / 47	47 / 47	47 / 47	47 / 47	46 / 47
Number of 3D points	9,091	8,159	11,441	8,852	7,517
Mean re-projection error	1.047 pixels	0.855 pixels	1.020 pixels	0.890 pixels	0.893 pixels
Point cloud size (memory gain)	27.02 MB (1×)	19.00 MB (1.42×)	12.20 MB (2.22×)	3.50 MB (7.72×)	3.22 MB (8.39×)
Elapsed time (performance gain)	922 *sec* (1×)	59.522 *sec* (15.490×)	57.249 *sec* (16.105×)	68.164 *sec* (13.526×)	76.288 *sec* (12.086×)

Table 4.11 Performance comparison of 3D reconstruction for “ikea” data set

Package	**Bundler**	**HD^{4}AR**			
Descriptor	**SIFT**	**SIFT**	**SURF**	**FREAK**	**BRISK**
Number of registered images	34 / 44	43 / 44	39 / 44	40 / 44	36 / 44
Number of 3D points	3,013	7,375	6,350	14,868	9,043
Mean re-projection error	2.308 pixels	0.781 pixels	1.284 pixels	0.788 pixels	0.790 pixels
Point cloud size (memory gain)	24.69 MB (1×)	16.30 MB (1.52×)	5.98 MB (4.13×)	5.35 MB (4.62×)	3.37 MB (7.33×)
Elapsed time (performance gain)	1,533 *sec* (1×)	98.420 *sec* (15.576×)	145.863 *sec* (10.510×)	167.802 *sec* (9.136×)	126.222 *sec* (12.145×)

Contrast to outdoor data sets, binary descriptors worked better than the SIFT descriptor for all indoor data sets except the "engine" data set, in terms of metrics presented in Tables 4.8-4.11, e.g., the number of 3D points, point cloud size, elapsed time, etc. The "engine" data set was a photo collection from an actual user who was a beginner to use SfM-based 3D reconstruction. The overlapping portion between the images in the "engine" data set was relatively low, i.e., 5-10%, and therefore, it was difficult to register an entire image set using the proposed 3D reconstruction algorithm. Nevertheless, the HD^{4}AR with the SIFT descriptor successfully registers almost every image. The HD^{4}AR with other descriptors also produced more dense point clouds compared to the Bundler and resulted smaller mean re-projection errors. In addition, the generated physical models for the "engine" data set were able to provide mobile augmented reality services with the proposed model-based 6-DOF localization method. The details of localization results will be discussed in Section 4.4.2. Figure 4.14 shows the generated 3D physical models from all indoor data sets using the FREAK descriptor

Based on experimental results discussed in this section, we illustrate the potential of the HD^{4}AR 3D reconstruction for rapidly creating 3D point clouds from real-world data sets. Due to enhancements presented in Section 4.2, such as combination of binary feature descriptor, post-filtering during the SfM, and hardware/software parallelism, the HD^{4}AR took 1-3 *min* to generate a 3D point cloud for indoor images and 5-20 *min* for outdoor images with binary descriptors. Compared to the Bundler, the most widely-used SfM package using an incremental approach, the HD^{4}AR achieved the performance gain up to

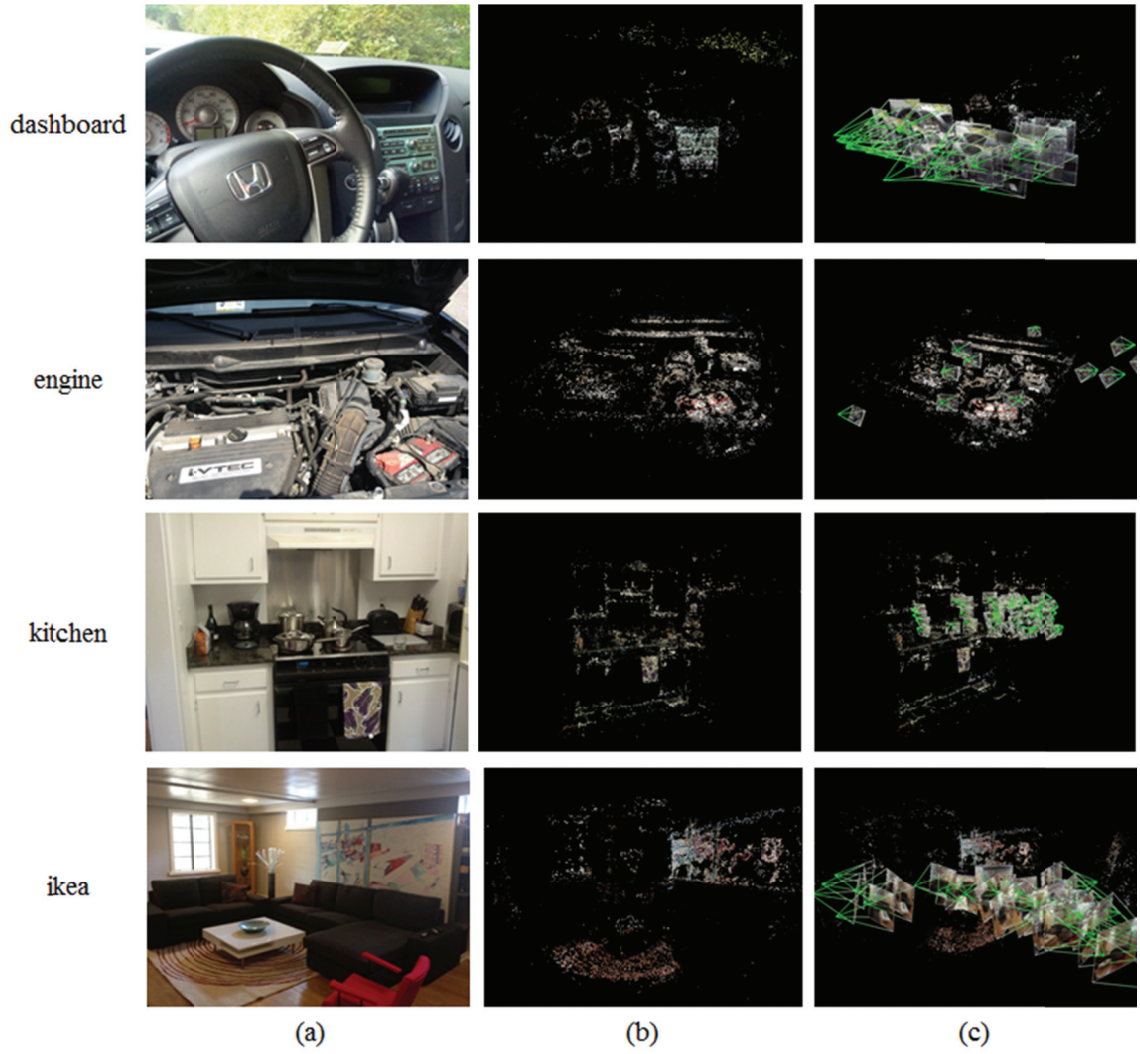

Figure 4.14 3D reconstruction results for room-scale indoor data sets with FREAK descriptor: (a) initial base images, (b) 3D point clouds from the HD^{4}AR, and (c) 3D point clouds with estimated camera position of input base images

2,875%. By considering all the results shown in Tables 4.2-4.11, we can conclude that the proposed parallelized SfM approach works well with both indoor and outdoor data sets, i.e., from room-level to street-level scales, and achieves significant gains on both speed and accuracy compared to existing work. The binary feature descriptors, such as FREAK and

BRISK, are appropriate for fast 3D reconstruction and still generate accurate 3D point clouds with less memory consumption. Furthermore, the HD^{4}AR successfully generates 3D point clouds purely based on images and does not require any constraints on photographs, such as geo-tag, ordered sequence, etc. In all cases, the maximum re-projection error is few image pixels, and therefore, generated 3D point clouds well-represent target scene and can be used for mobile augmented reality.

4.4.2 Model-based 6-DOF Localization/Augmentation

In order to measure the capability of model-based localization with generated 3D physical models, the localization tests were performed on each 3D physical model. All the photos were newly taken by smartphones, such as Apple iPhone 4S and Samsung Galaxy Nexus, at random location. A group of images were tested for on spot localization using the client-server architecture and 4G LTE connections to assess the mobility of the HD^{4}AR.

In this experiment, we measured the localization performance with the sequential requests from a single device as well as with the multiple simultaneous requests of localization from several client devices. This is particularly important as the HD^{4}AR server can handle parallel localization requests from client devices simultaneously, which leads to the increased system capacity. For example, if two users attempt to query cyber-information at the same time by submitting two separate localization requests, both user requests will be processed simultaneously and augmentation results will be presented within the same 1-2 *sec* time span. Considering the number of smart devices these days, this feature shows the

scalability in implementing the proposed solution for near real-time exchange of information among numerous users.

The performance of the Bundler package was measured and compared to that of the HD^{4}AR to demonstrate the performance gains on localization. Since the Bundler package does not provide on spot localization and cyber-information association, we only compared the offline localization speed of the Bundler to that of the HD^{4}AR. To demonstrate the augmentation capability of the HD^{4}AR, 3D cyber-information is pre-associated to the 3D physical models using the 3D content authoring method proposed in this study. The proposed approach for 3D cyber-physical content authoring will be fully discussed in Chapter 5. During the localization/augmentation experiment, following metrics were measured:

- *Localization success-ratio*: how many new photographs are successfully localized. Due to the model-based localization approach, the success in localization means that the system was able to solve the camera calibration equation, i.e., Direct Linear Transformation equation followed by a Levenberg-Marquardt optimization, using given 2D-to-3D correspondences between image and 3D physical model.
- *Mean number of 2D-to-3D matches*: average number of 2D-to-3D correspondences found in a single photograph using the proposed direct 2D-to-3D matching algorithm. The found correspondences are used in the camera calibration equation to estimate a complete pose of the camera. Due to the limitation of the Bundler

package, this metric cannot be measured for the Bundler.

- *Mean re-projection error*: overall mean re-projection error that computed by projecting each 3D point into the localized photographs. Specifically, the re-projection error is the distance between projected 3D points and original image feature points in 2D-to-3D matching results. The value of this metric indicates the accuracy of localization. This metric is also not measured for the Bundler.
- *Mean localization time (sequential requests)*: how long does it take to localize a single photograph on average with sequential requests from a single device. The localization time consists of feature detection/extraction time, direct 2D-to-3D matching time, and camera calibration time.
- *Mean localization time (parallel requests)*: how long does it take to localize a single photograph on average with parallel requests from multiple devices. Specifically, the HD^{4}AR server runs sixteen parallel threads for localization where each thread can handle a single photograph at a time. Since the Bundler does not support parallel processing, this metric cannot be measured for the Bundler.

Tables 4.12-4.14 compare the overall results of model-based 6-DOF localization on the 3D physical models of outdoor buildings, i.e., “patton”, “knu”, and “parliament” models. The proposed direct 2D-to-3D matching with 3D representative descriptors achieved the significant performance gain even with sequential localizations. In all cases, the HD^{4}AR rapidly localized photographs submitted by client devices, and was 1,960-11,533% faster than the Bundler. The HD^{4}AR was about 20 times faster than the Bundler even with the

same SIFT descriptor due to the proposed direct 2D-to-3D matching. As we outlined in Section 4.3, the Bundler does the 2D-to-2D-to-3D matching and compares the newly submitted image to an entire set of base images.

Within the HD^{4}AR system, the SIFT descriptor produced the minimum mean re-projection error, which means the most accurate localization, but it was significantly slower than other descriptors, i.e., the SURF, FREAK, and BRISK, due to time consuming computations and a twice longer dimension of descriptors. On the other hand, the SURF descriptor enabled fast localizations, but caused the most erroneous results among the tested descriptors. As we will discuss throughout this section, the mean re-projection errors from the SURF descriptor were even worse in the case of indoor localizations. Finally, the binary descriptors also achieved the significant performance gain compared to the Bundler and resulted mean re-projection errors in the range of 0.872-1.189 pixels. If we only focus on the binary descriptors, the performance gain compared to the Bundler was 9,791-11,081% for building-scale outdoor data sets.

One of the interesting measurements for localization is the mean re-projection error, which is represented in image pixel units. Since a camera is projecting an entire 3D scene in front of the camera into 2D image space, it is difficult to map this mean re-projection error into real-world distance metric, such as centimeters or millimeters. For example, a small pixel error will result the significant error in real-world if the subject is very far from the camera.

Table 4.12 Performance comparison of 6-DOF localization for “patton” models

Package	**Bundler**	**HD^{4}AR**			
Descriptor	**SIFT**	**SIFT**	**SURF**	**FREAK**	**BRISK**
Localization success-ratio	50 / 50 (100%)	49 / 50 (98%)	49 / 50 (98%)	50 / 50 (100%)	49 / 50 (98%)
Mean number of 2D-to-3D matches	-	9,143	3,576	2,146	2,145
Mean re-projection error	-	0.627 pixels	0.895 pixels	0.872 pixels	0.812 pixels
Mean localization time (sequential requests)	242.775 *sec* (1×)	12.389 *sec* (19.596×)	2.105 *sec* (115.333×)	2.191 *sec* (110.806×)	2.312 *sec* (105.006×)
Mean localization time (parallel requests)	-	3.527 *sec* (-)	0.663 *sec* (-)	0.514 *sec* (-)	0.754 *sec* (-)

Table 4.13 Performance comparison of 6-DOF localization for “knu” models

Package	**Bundler**	**HD^{4}AR**			
Descriptor	**SIFT**	**SIFT**	**SURF**	**FREAK**	**BRISK**
Localization success-ratio	50 / 50 (100%)	50 / 50 (100%)	49 / 50 (98%)	50 / 50 (100%)	50 / 50 (100%)
Mean number of 2D-to-3D matches	-	2,258	1,521	1,241	1,204
Mean re-projection error	-	0.808 pixels	1.300 pixels	1.189 pixels	1.070 pixels
Mean localization time (sequential requests)	120.820 *sec* (1×)	6.057 *sec* (19.947×)	1.173 *sec* (103.001×)	1.234 *sec* (97.909×)	1.347 *sec* (89.696×)
Mean localization time (parallel requests)	-	1.600 *sec* (-)	0.369 *sec* (-)	0.346 *sec* (-)	0.507 *sec* (-)

Table 4.14 Performance comparison of 6-DOF localization for "parliament" models

Package	**Bundler**[(a)]	**HD^{4}AR**			
Descriptor	**SIFT**	**SIFT**	**SURF**	**FREAK**	**BRISK**
Localization success-ratio	-	40 / 40 (100%)	40 / 40 (100%)	40 / 40 (100%)	40 / 40 (100%)
Mean number of 2D-to-3D matches	-	6,362	670	449	465
Mean re-projection error	-	0.613 pixels	1.226 pixels	0.928 pixels	0.897 pixels
Mean localization time (sequential requests)	-	6.193 *sec* (×)	1.831 *sec* (×)	2.391 *sec* (×)	2.693 *sec* (×)
Mean localization time (parallel requests)	-	2.684 *sec* (-)	0.784 *sec* (-)	0.768 *sec* (-)	0.847 *sec* (-)

[(a)] The Bundler failed to create 3D point cloud due to image size and out of memory problem.

As a consequence, the distance from camera to target subject must be considered when converting a mean re-projection error into a real-world distance metric:

$$e_{mm} = \frac{e_{pixel}}{f_{pixel}} \cdot d_{mm} = \frac{e_{pixel}}{w_{pixel}} \cdot \frac{w_{mm}}{f_{mm}} \cdot d_{mm} \tag{4.6}$$

where e_{mm} is a real-world distance error in millimeter unit, e_{pixel} is a localization re-projection error in pixel units, f_{pixel} is a focal length in pixel unit, d_{mm} is a distance from camera center to target subject in millimeter unit, w_{pixel} is an image width in pixel units, and w_{mm} and f_{mm} are a camera CCD sensor width and a focal length in millimeter unit, respectively. For example, by using an Equation 4.6 and the camera parameters of Apple

iPhone 4S, i.e., $w_{mm} = 4.54$, $f_{mm} = 4.28$, $w_{pixel} = 3{,}264$, the experimental results shown in Tables 4.12-4.14 can be interpreted as the HD^{4}AR localization had 1.992-4.225 *mm* error if Apple iPhone 4S was used to take a picture and the subject was 10 meters away from the camera .

Figure 4.14 shows the example of localization/augmentation results from the HD^{4}AR with the BRISK descriptor for building-scale outdoor images. The 3D physical models generated from the HD^{4}AR were fed into a multi-view stereo algorithm [52, 53] to increase the density of point clouds for visualization purposes. The generated dense point clouds were not used for the localization and only for visualizing the models to end-users. The dense 3D physical models associated with 3D cyber-information are shown in Figure 4.15a. Figure 4.15b illustrates the HD^{4}AR localization results in 3D space and corresponding augmented photographs are shown in Figure 4.14c. In addition to experimental results shown in Tables 4.12-4.14, the augmented photographs empirically show that camera poses were successfully recovered, and thus the cyber-information, e.g., window information on the "patton" model, is precisely overlaid on photographs from different viewpoints.

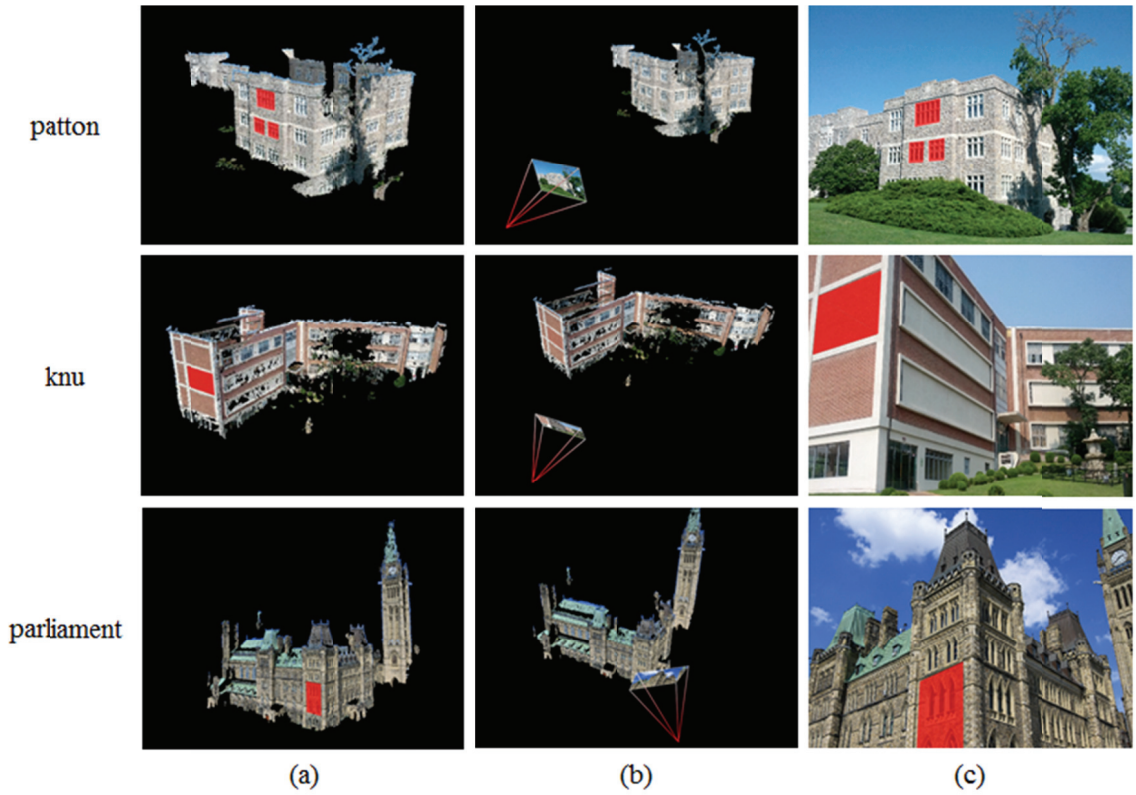

Figure 4.15 Localization/Augmentation results for building-scale outdoor data sets: (a) Target 3D model associated with 3D cyber-information, (b) 6-DOF localization result from the HD^{4}AR server, and (c) Augmentation results from the HD^{4}AR client

Tables 4.15-4.17 compare the overall results of model-based 6-DOF localization on the outdoor construction jobsite models, i.e., "rtfr", "cfta", and "rh" models. Again, the HD^{4}AR outperformed the Bundler and was 2,518-16,221% faster for outdoor construction data sets. The localization results for construction jobsites clearly show the strength of the proposed direct 2D-to-3D matching algorithm. For "rtfr", "cfta", and "rh" models, the number of base images used for 3D reconstruction was relatively large compared to other data sets, as shown in Table 4.1. Consequently, the Bundler took much longer time for localizing a photograph as it performs 2D-to-2D-to3D matching. On the contrast, the HD^{4}AR only

compares the image descriptors to 3D representative descriptors, and thus, the elapsed time only depends on the size of the physical model, resulting significant performance gain. With the binary descriptors, the HD^{4}AR was up to 160 times faster than the Bundler and the mean re-projection errors were between 1.000-2.511 pixels. The localization error is slightly higher than the building-scale outdoor cases, but is still in the range of few image pixels. Figure 4.16 shows the example of localization/augmentation results from the HD^{4}AR with the BRISK descriptor for construction jobsite photographs. The augmented photographs show that the HD^{4}AR precisely delivered/visualized associated cyber-information in street-scale outdoor environment.

Another interesting measurement for localization is the mean number of 2D-to-3D matches. This measurement indicates the number of found correspondences between image feature points and 3D points in a single photograph. As shown in Tables 4.2-4.7 and 4.12-4.17, the measured numbers of 2D-to-3D matches is much smaller than the number of 3D points in the 3D physical models. This might be due to the fact that submitted photographs from the client devices only cover the part of the target scene, experience different illumination conditions, or are low quality photographs caused by camera shake. Nevertheless, the HD^{4}AR accurately and rapidly localized the submitted photograph with small number of 2D-to-3D correspondences, and this fact leads to a cached *k-d* tree approach to further accelerate the direct 2D-to-3D matching algorithm. By caching and maintaining highly queried 3D points in the small memory, we can further reduce the localization time. The details of a cached *k-d* tree approach will be fully discussed in Chapter 6.

Table 4.15 Performance comparison of 6-DOF localization for “rtfr” models

Package	**Bundler**	**HD^{4}AR**			
Descriptor	**SIFT**	**SIFT**	**SURF**	**FREAK**	**BRISK**
Localization success-ratio	49 / 50 (98%)	49 / 50 (98%)	46 / 50 (92%)	49 / 50 (98%)	50 / 50 (100%)
Mean number of 2D-to-3D matches	-	907	1,300	1,692	1,637
Mean re-projection error	-	1.969 pixels	2.702 pixels	2.511 pixels	2.435 pixels
Mean localization time (sequential requests)	177.725 *sec* (1×)	6.190 *sec* (28.712×)	2.214 *sec* (80.273×)	2.847 *sec* (62.425×)	3.059 *sec* (58.099×)
Mean localization time (parallel requests)	-	1.594 *sec* (-)	0.707 *sec* (-)	0.904 *sec* (-)	0.811 *sec* (-)

Table 4.16 Performance comparison of 6-DOF localization for “cfta” models

Package	**Bundler**	**HD^{4}AR**			
Descriptor	**SIFT**	**SIFT**	**SURF**	**FREAK**	**BRISK**
Localization success-ratio	50 / 50 (100%)	50 / 50 (100%)	50 / 50 (100%)	50 / 50 (100%)	50 / 50 (100%)
Mean number of 2D-to-3D matches	-	621	328	555	695
Mean re-projection error	-	0.696 pixels	1.697 pixels	1.189 pixels	1.000 pixel
Mean localization time (sequential requests)	72.488 *sec* (1×)	2.879 *sec* (25.178×)	0.795 *sec* (91.180×)	0.857 *sec* (84.583×)	0.856 *sec* (84.682×)
Mean localization time (parallel requests)	-	0.732 *sec* (-)	0.215 *sec* (-)	0.184 *sec* (-)	0.188 *sec* (-)

Table 4.17 Performance comparison of 6-DOF localization for "rh" models

Package	**Bundler**	**HD^{4}AR**			
Descriptor	**SIFT**	**SIFT**	**SURF**	**FREAK**	**BRISK**
Localization success-ratio	50 / 50 (100%)	50 / 50 (100%)	50 / 50 (100%)	50 / 50 (100%)	50 / 50 (100%)
Mean number of 2D-to-3D matches	-	467	495	470	601
Mean re-projection error	-	0.886 pixels	1.584 pixels	1.102 pixels	1.466 pixels
Mean localization time (sequential requests)	122.467 *sec* (1×)	2.467 *sec* (49.642×)	0.755 *sec* (162.208×)	0.765 *sec* (160.088×)	0.914 *sec* (133.990×)
Mean localization time (parallel requests)	-	1.026 *sec* (-)	0.321 *sec* (-)	0.254 *sec* (-)	0.291 *sec* (-)

Finally, Tables 4.18-4.21 compare the localization results for indoor scenarios. For the indoor test cases, i.e., "dashboard", "engine", "kitchen", and "ikea" models, the HD^{4}AR was 708-4,704% faster than the Bundler. Since the indoor images are typically texture-less and results less number of feature descriptors compared to outdoor images, the performance gain from feature descriptors is slightly reduced. However, the HD^{4}AR with the SURF, FREAK, and BIRSK descriptors are still 2,321-4,704% faster than the Bundler and took at most 2 *sec* to localize a single photograph. Due to the proposed direct 2D-to-3D matching, the HD^{4}AR with the SIFT descriptor was also 708-1,206% faster than the Bundler.

Especially, for "engine" and "ikea" data sets, the Bundler failed to localize all tested photographs. As shown in Tables 4.9 and 4.11, the Bundler produced the higher mean re-projection error in 3D reconstruction for "engine" and "ikea" data sets, and the resulting 3D

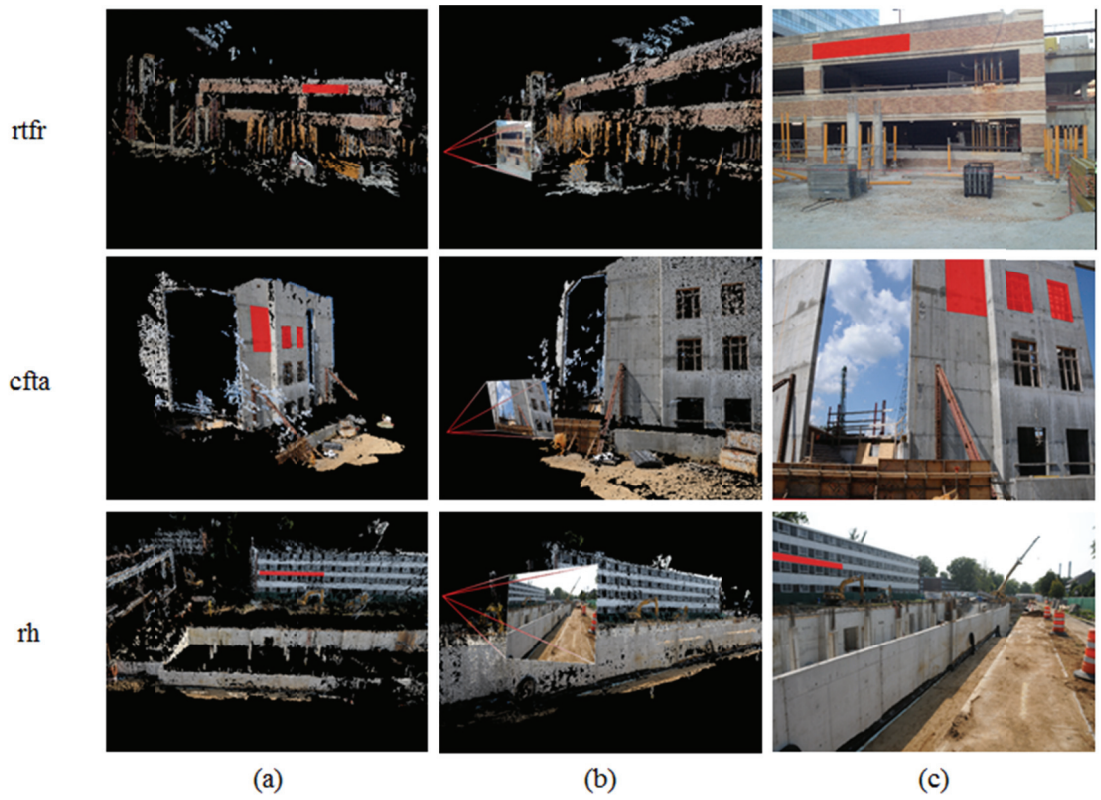

Figure 4.16 Localization/Augmentation results for street-scale construction jobsites: (a) Target 3D model associated with 3D cyber-information, (b) 6-DOF localization result from the HD^{4}AR server, and (c) Augmentation results from the HD^{4}AR client

point clouds from the Bundler did not work well for model-based localization. However, as shown in Tables 4.19 and 4.21, the HD^{4}AR was able to provide localization/augmentation results on those data sets although the SURF descriptor has the worst localization success-ratio and mean re-projection error among all tested descriptors. Figure 4.17 shows the localization and augmentation results for indoor data sets using the FREAK descriptor.

Table 4.18 Performance comparison of 6-DOF localization for “dashboard” models

Package	**Bundler**	**HD^{4}AR**			
Descriptor	**SIFT**	**SIFT**	**SURF**	**FREAK**	**BRISK**
Localization success-ratio	40 / 40 (100%)	39 / 40 (97.5%)	40 / 40 (100%)	40 / 40 (100%)	40 / 40 (100%)
Mean number of 2D-to-3D matches	-	381	391	486	375
Mean re-projection error	-	1.250 pixels	2.514 pixels	1.909 pixels	1.947 pixels
Mean localization time (sequential requests)	34.407 *sec* (1×)	3.432 *sec* (10.025×)	0.794 *sec* (43.334×)	0.907 *sec* (37.935×)	0.930 *sec* (36.997×)
Mean localization time (parallel requests)	-	0.928 *sec* (-)	0.344 *sec* (-)	0.312 *sec* (-)	0.353 *sec* (-)

Table 4.19 Performance comparison of 6-DOF localization for “engine” models

Package	**Bundler**	**HD^{4}AR**			
Descriptor	**SIFT**	**SIFT**	**SURF**	**FREAK**	**BRISK**
Localization success-ratio	0 / 45 (0%)	45 / 45 (100%)	42 / 45 (93.3%)	41 / 45 (91.1%)	41 / 45 (91.1%)
Mean number of 2D-to-3D matches	-	2,717	2,611	1,232	1,162
Mean re-projection error	-	1.213 pixels	1.642 pixels	2.033 pixels	2.200 pixels
Mean localization time (sequential requests)	49.667 *sec* (1×)	7.016 *sec* (7.079×)	2.140 *sec* (23.209×)	1.950 *sec* (25.470×)	2.282 *sec* (21.765×)
Mean localization time (parallel requests)	-	1.868 *sec* (-)	0.569 *sec* (×)	0.534 *sec* (×)	0.616 *sec* (×)

Table 4.20 Performance comparison of 6-DOF localization for “kitchen” models

Package	**Bundler**	**HD^{4}AR**			
Descriptor	**SIFT**	**SIFT**	**SURF**	**FREAK**	**BRISK**
Localization success-ratio	50 / 50 (100%)	49 / 50 (98%)	49 / 50 (98%)	50 / 50 (100%)	50 / 50 (100%)
Mean number of 2D-to-3D matches	-	457	487	411	373
Mean re-projection error	-	1.149 pixels	1.981 pixels	1.766 pixels	1.748 pixels
Mean localization time (sequential requests)	23.894 *sec* (1×)	1.981 *sec* (12.062×)	0.508 *sec* (47.035×)	0.496 *sec* (48.173×)	0.529 *sec* (45.168×)
Mean localization time (parallel requests)	-	0.547 *sec* (-)	0.176 *sec* (-)	0.167 *sec* (-)	0.174 *sec* (-)

Table 4.21 Performance comparison of 6-DOF localization for “ikea” models

Package	**Bundler**	**HD^{4}AR**			
Descriptor	**SIFT**	**SIFT**	**SURF**	**FREAK**	**BRISK**
Localization success-ratio	0 / 45 (0%)	44 / 45 (97.8%)	31 / 45 (68.9%)	44 / 45 (97.8%)	43 / 45 (95.6%)
Mean number of 2D-to-3D matches	-	450	227	714	480
Mean re-projection error	-	1.394 pixels	3.801 pixels	2.301 pixels	2.416 pixels
Mean localization time (sequential requests)	38.955 *sec* (1×)	4.356 *sec* (8.943×)	0.914 *sec* (42.62×)	1.022 *sec* (38.116×)	1.115 *sec* (34.937×)
Mean localization time (parallel requests)	-	1.188 *sec* (-)	0.270 *sec* (-)	0.251 *sec* (-)	0.262 *sec* (-)

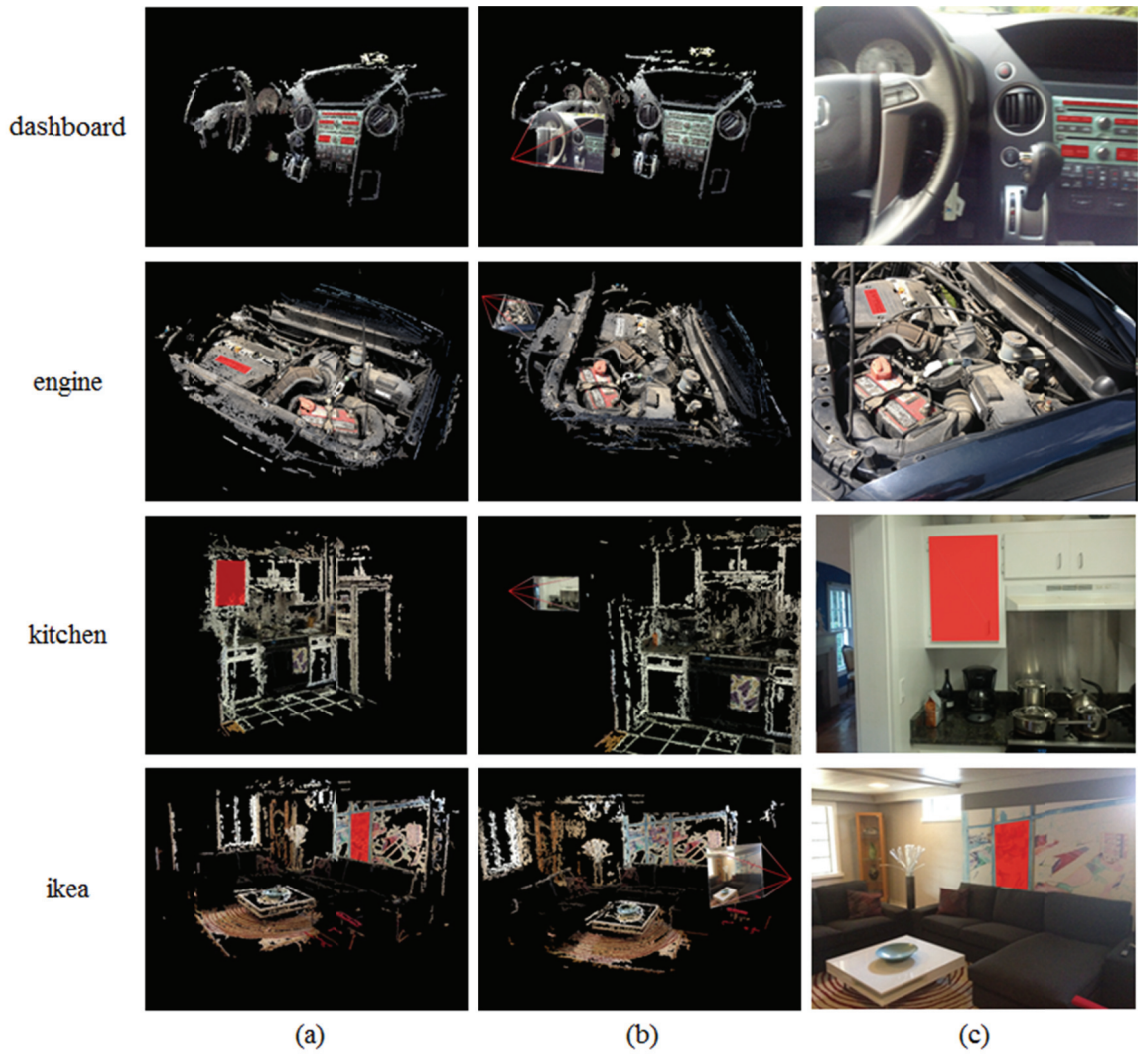

Figure 4.17 Localization/Augmentation results for room-scale indoor data sets: (a) Target 3D model associated with 3D cyber-information, (b) 6-DOF localization result from the HD^{4}AR server, and (c) Augmentation results from the HD^{4}AR client

In this section, we illustrate the potential of the HD^{4}AR for successfully providing high-precision mobile augmented reality using model-based localization. The HD^{4}AR localizes a given photograph solely based on the content of images captured by mobile devices and does not require any sensors or infrastructures for localization. Furthermore, it has a

capacity to provide high-precision localization with maximum error up to few image pixels. With the HD^{4}AR and the binary descriptors, the localization/augmentation of single image took about 0.5-3.0 *sec* in all cases. Thus, the HD^{4}AR can provide near real-time localization and augmentation capabilities for both indoor and outdoor environments.

4.5 Contributions and Significance

The proposed HD^{4}AR approach, a vision-based marker-less method using SfM-based 3D point cloud models, was designed with the intent of bringing high-precision mobile augmented reality to end-users without requiring external sensors or infrastructures. As a consequence, the HD^{4}AR promises the applicability of model-based localization on the field of high-precision mobile augmented reality. The HD^{4}AR rapidly generates a 3D point cloud model, which roles as a reference model for localization, and provides near real-time, high-precision localization and augmentation solely based on the photograph. The experimental results shown in Section 4.4 indicate the robustness of the system to dynamic changes of viewpoint, camera resolution, and scale of objects, which are typically observed in many practical mobile augmented reality applications.

Based on discussion in this Chapter, we can conclude that the HD^{4}AR – hybrid mobile/cloud model-based localization on SfM-based 3D physical model – has successfully filled the “Research Gap 1: Fine-grained 6-DOF Localization with Mobile Devices”, and “Research Gap 3: Near Real-time Cyber-physical Information Association at Dynamically Varying Environmental Scales”. Table 4.22 compares the proposed approach with all

related works reviewed in Section 3.1. The proposed approach purely localizes users based on images from mobile devices and works well for both indoor and outdoor environment without requiring any sensors or infrastructures for localization. Also a bootstrapping process of the system is significantly accelerated by proposing a new parallelized 3D reconstruction pipeline. The HD^{4}AR provides high-precision 6-DOF localization where uncertainty level is 0.613-2.511 pixels and near real-time localization/augmentation, which takes 0.5-3.0 *sec* to localize a single image. Through the extensive experiments, we also proved that the binary descriptors work well for both 3D reconstruction and model-based 6-DOF localization. Finally, the proposed approach successfully supports on spot localization through the client-server architecture and is scalable for multi-user scenarios.

Table 4.22 Validation of the HD^{4}AR approach

Metrics	**Sensor-based**	**Marker-based**	**Visual SLAM**	**Model-based**	**HD^{4}AR**
Localization Accuracy	1.5 – 35 *m* [(a)]	0.5 – 2 *mm* [(b)]	0.5 – 20 *mm* [(c)]	0.5 – 20 *mm* [(c)]	2 – 8 *mm* [(c)]
Localization Speed	100 – 200 *msec*	20 – 140 *msec*	20 – 40 *msec*	5 – 240 *sec*	0.5 - 3.0 *sec*
External Infrastructure	GPS satellite	Optical markers	Not needed	Not needed	Not needed
Resistant to drifts and error accumulation	×	✓	×	✓	✓
Scale well to large scene	✓	×	×	✓	✓

[(a)] GPS Covered area; [(b)] Markers within 3*m* distance; [(c)] Objects within 10*m* distance.

5 Plane Transformation based 3D Cyber-physical Content Authoring from A Single 2D Image

5.1 Overview of Solution Approach to Research Gap 2

As discussed in Sections 2.2 and 3.2, the mobile augmented reality system should provide a way of making cyber-information and associating it with real-world physical objects so that other users can see generated cyber-information overlaid on top of corresponding objects in the photograph. For high-precision mobile augmented reality, which provides 6-DOF localization in 3D space, all deliverable cyber-information should also have 3D positional information so that the cyber-information can be properly projected in to the photograph with the recovered 6-DOF pose of a camera.

The most straightforward method for this 3D content authoring is preparing a 3D drawing of target object or building and manually aligning it to physical objects [1], as shown in Figure 5.1. Although this approach can deliver a plenty of information to end users, it always require manual association and a 3D drawing generated by specific 3D design frameworks, such as CAD tools. However, the question of how to conveniently and accurately create even simple 3D content using a mobile device and 2D interface is still an open problem [19].

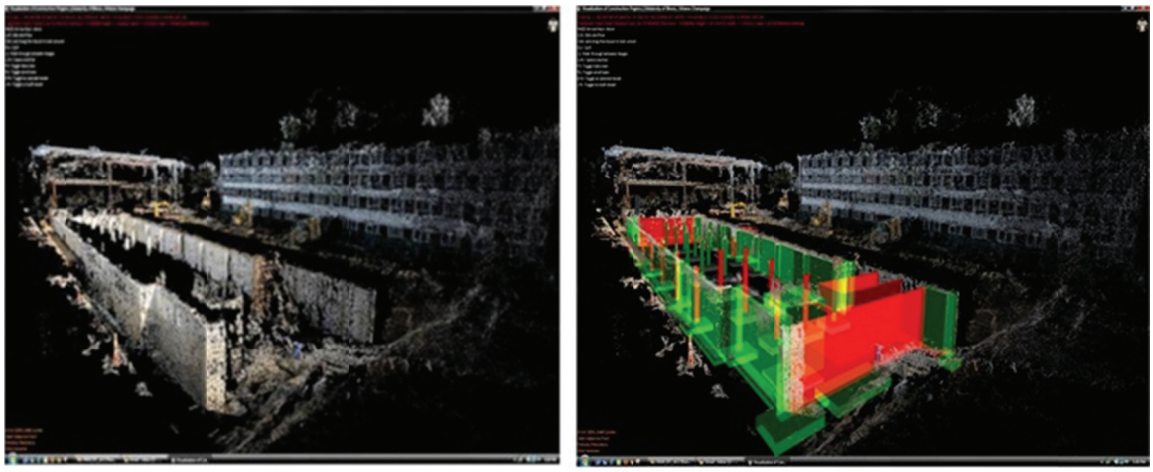

Figure 5.1 An example of 3D cyber-physical model: (a) 3D point cloud of construction site, (b) 3D building plan model aligned with the point cloud (Adopted from [1])

Therefore, a new approach, which can create 3D cyber-contents and associate them with the 3D physical objects using a single 2D image, is developed to fill the "Research Gap 2: 3D Cyber-physical Content Authoring from 2D Interface". With this approach, a user can easily create and associate new 3D cyber-information by simply drawing a polygon on the photograph, and thus can work with commodity smartphones which typically have 2D user interfaces.

To enable 3D content authoring from a single 2D image, the proposed approach makes use of 1) plane image transformation to automatically find 2D correspondences of user inputs on other images and 2) camera parameters, such as focal length, radial distortion coefficient, rotational and translational matrix, recovered during the HD^{4}AR 3D reconstruction, to accurately triangulate 2D user inputs and derive 3D positional information of them. The

details of the proposed approach, i.e., “Plane Transformation based 3D Cyber-physical Content Authoring from A Single 2D image”, will be discussed in the following section.

5.2 3D Content Authoring with Homography

The proposed 3D content authoring method from a single 2D image is based on plane image transformation, i.e., a homography matrix. By its definition, the homography is an invertible transformation in a projective space that maps an image plane to another image plane. For example, each pixel in image plane #1 can be transformed to another image plane #2 via homography matrix:

$$s\begin{bmatrix} x_2 \\ y_2 \\ 1 \end{bmatrix} = \mathbf{H}\begin{bmatrix} x_1 \\ y_1 \\ 1 \end{bmatrix} \tag{5.1}$$

where **H** is an estimated 3×3 homography matrix, (x_1, y_1) is a pixel coordinates in image plane #1, and (x_2, y_2) is a transformed pixel coordinates of (x_1, y_1) in the image plane #2. As shown in Figure 5.2, one image plane can be accurately transformed to another image plane using estimated homography matrix. The homography matrix between two images can be automatically found using the RANSAC with normalized Direct Linear Transform algorithm [48], as discussed in Section 4.2.2. By using Equation 5.1 and the estimated homography matrix, we can find the correspondences of 2D points between two images.

Since the HD^{4}AR 3D reconstruction discussed in Section 4.2 estimates homography matrices between every base image pair and keeps those matrices in the 3D physical model,

Figure 5.2 Homography transformation: (a) image 1, (b) image 2, (c) image 1 is transformed to image plane 2 using estimated homography matrix

we can utilize these homography matrices to find correspondences of a user-created 2D element on each base image. For example, windows drawn by the user can be correctly found on other base images using estimated homography matrices, as illustrated in Figure 5.3a and 5.3b. To increase the accuracy of found correspondences, we only investigate base images which *H-Score* is greater than 0.85. As outlined in Section 4.2, the *H-Score* is the percentage of number of inliers during the feature matching stage with estimated homography matrix. The higher *H-score* means that two images have many matches survived after fitting into the homography matrix, and therefore the estimated homogrpahy matrix accurately describes the plane transformation between those images.

Using this 2D correspondence information as well as intrinsic and extrinsic camera parameters recovered during 3D reconstruction, our method then triangulates each vertex of the user-created polygon to impose 3D positional information to user-created 2D element. If the estimated 2D correspondences of user-created element are not located within the image dimension of the base image, that correspondence information is discarded for triangulation. In addition, if the recovered camera parameters of the base image had a mean re-projection error higher than 1.0 pixel during the 3D reconstruction, that base image is also discarded for triangulation. With these constraints, we found that 3-8 base cameras were typically participated in the triangulation. The experimental results of 3D content authoring will be discussed in Section 5.3.

The polynomial method [50] is used for triangulation to handle the noise presented in user measurements and automatically found 2D correspondences. After fixing camera parameters and running Bundle Adjustment to further minimize a mean re-projection error of the triangulated polygon, the resulting 3D element is well-aligned with the existing 3D physical model as shown in Figure 5.3c. Once this user-created element has 3D positional information, it can be precisely overlaid on other photographs from different viewpoints using the HD^{4}AR model-based localization, as shown in Figure 5.3d.

This simple and robust 3D cyber-physical content authoring method based on homography can help users create 3D cyber-information easily by drawing a simple polygon on a single 2D image. In addition, the proposed approach automatically associates user-created cyber-

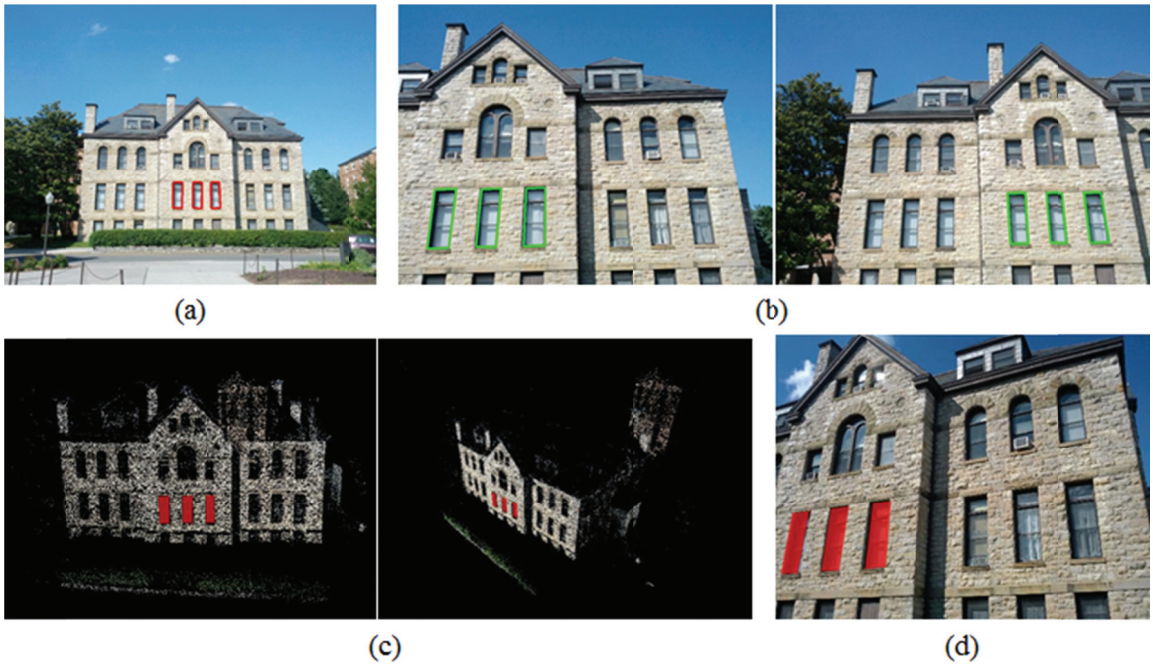

Figure 5.3 The proposed 3D cyber-physical content authoring method: (a) A user marks windows on the photograph, (b) Using the estimated homographies, the system automatically finds correspondences of windows for each base image, (c) The system triangulates window elements using camera information of base images (which is recovered during the 3D reconstruction), (d) Mobile augmented reality: user-created window contents can be precisely overlaid on other photographs from different viewpoint.

information with the underlying 3D physical model, and therefore, users do not have to manually positioning and associating 3D cyber-information in 3D geometry. As a consequence, the proposed approach can be used in any commodity smartphones which typically have a capability of showing an image on their displays and tracking user's touch points to draw the polygon. Figure 5.4 shows an example of 3D cyber-physical models, i.e., 3D cyber-information associated with 3D physical models, generated from the proposed method.

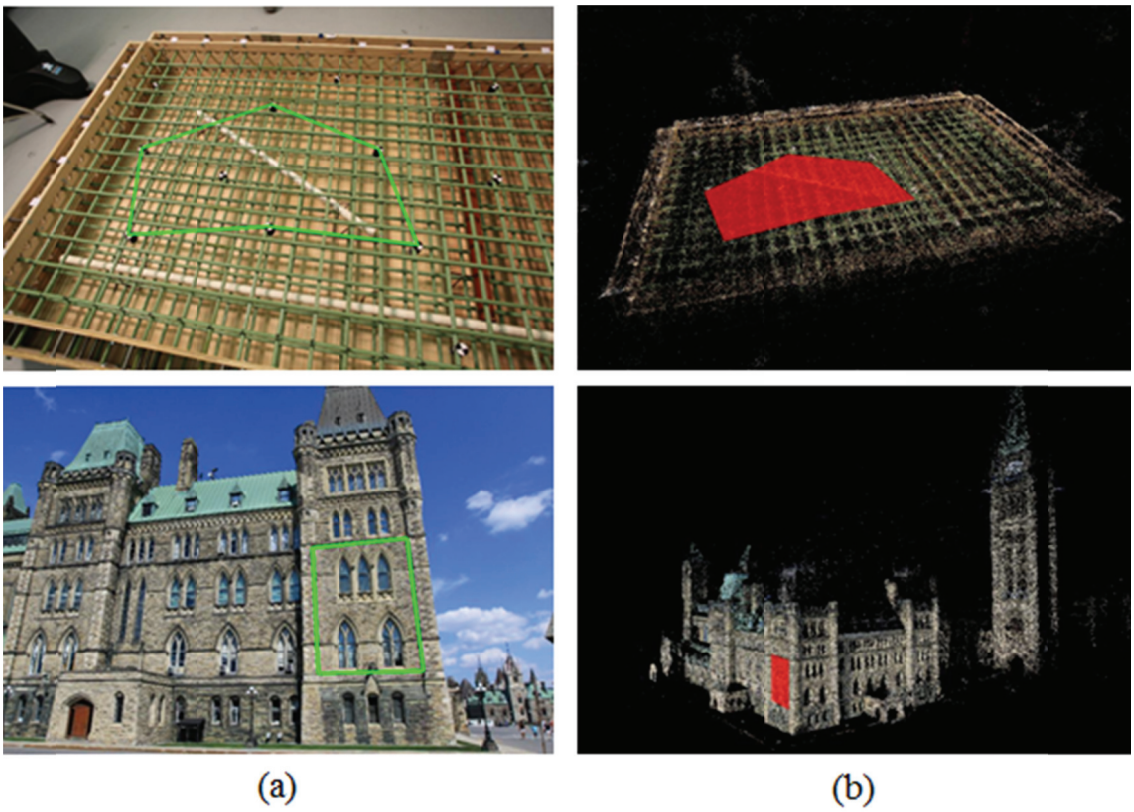

Figure 5.4 3D cyber-physical models from the proposed method: (a) user-input on a 2D image, (b) Generated cyber-information in 3D geometry: 3D cyber-information is well-aligned to 3D physical models.

5.3 Experimental Results and Validation

This section presents experimental results and the validation of the proposed 3D content authoring method. Since it is impractical to measure the ground truth position of every physical object on the 3D point cloud model, which often consists of sparse 3D points, we focused on demonstrating the capability of generating 3D cyber-information from 2D interface using commodity smartphones and empirically made a decision whether cyber-information was accurately associated with physical objects or not. In addition to visual

analysis, however, we also measured the mean re-projection error of triangulated 3D elements against the base images that were participated in triangulation. All the experiments were conducted on a single Amazon EC2 instance server with 22.5 GB memory and two Intel Xeon X5570 processors running Ubuntu version 12.04.

The experiment for 3D cyber-physical content authoring is performed in following procedure: 1) let users draw polygons on interesting objects on the single image with smartphones, 2) perform the proposed content authoring method and visualize generated 3D cyber-information with 3D point cloud model to see the accuracy of 3D cyber-information triangulation, and 3) test localization/augmentation on different location and viewpoint to verify that created 3D cyber-information is indeed well-associated in 3D geometry. The test tool for augmentation was based on the HD^{4}AR discussed in Chapter 4.

Table 5.1 shows the results of 3D cyber-physical content authoring with the proposed method. In all cases from indoor to outdoor data sets, the proposed method successfully generated 3D contents from user inputs on a single 2D image. During the estimation of 2D correspondences of user inputs on other base images using estimated homography matrices, we only used the base images which *H-score* is greater than 0.85 in order to increase the accuracy of triangulation. As a consequence, only 2-8 base images were participated in triangulation and the mean re-projection errors of the triangulated elements were in the range between 0.268-3.443 pixels. Figures 5.5-5.7 show the visual analysis results of the 3D cyber-physical content authoring with the proposed method. For example, a user drew

Table 5.1 3D cyber-physical content authoring results with 3D physical models generated by BRISK descriptor

Environment	Name	Number of vertices for user-driven elements	Number of base images participated in triangulation	Mean re-projection error
Outdoor	patton	15	8	2.619 pixels
	knu	4	4	0.268 pixels
	parliament	4	6	0.777 pixels
	rtfr	4	4	3.443 pixels
	cfta	12	6	1.464 pixels
	Rh	4	5	0.914 pixels
Indoor	dashboard	20	4	0.432 pixels
	engine	4	2	1.276 pixels
	kitchen	4	2	0.205 pixels
	ikea	4	3	0.686 pixels

several windows on “patton” image and the proposed method precisely triangulated and associated them with corresponding objects in the “patton” 3D physical model, as shown in Figure 5.5. Similarly, user-created cyber-buttons on “dashboard” image were successfully associated with the buttons in the “dashboard” 3D physical model, as shown in Figure 5.7. Once these user-created elements were successfully attached and aligned to 3D physical models, users can see this cyber-information precisely overlaid on the photograph taken from different locations and orientations (see Figures 5.5c, 5.6c, and 5.7c).

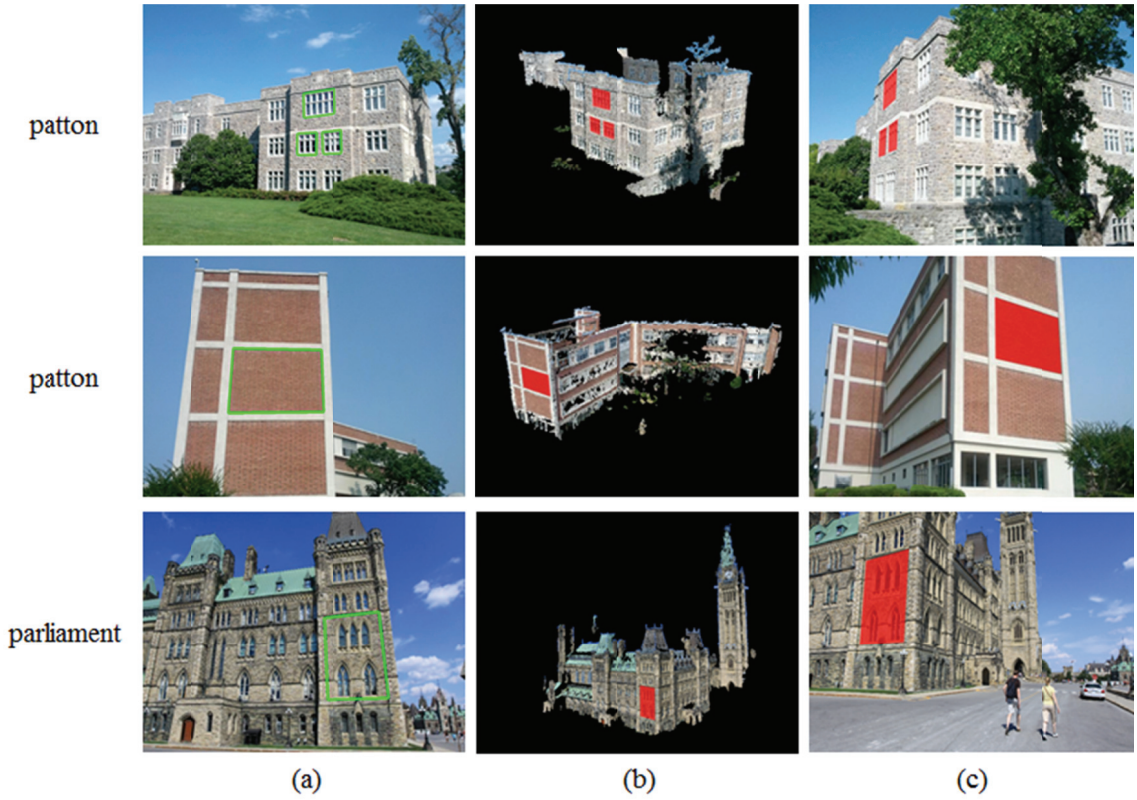

Figure 5.5 Results of 3D cyber-physical content authoring with the proposed method on building-scale outdoor data sets. (a) user-created information on the 2D image, (b) 3D elements driven from the user-created 2D elements, and (c) augmentation results of the user-created 3D cyber-information on another smart device on the site

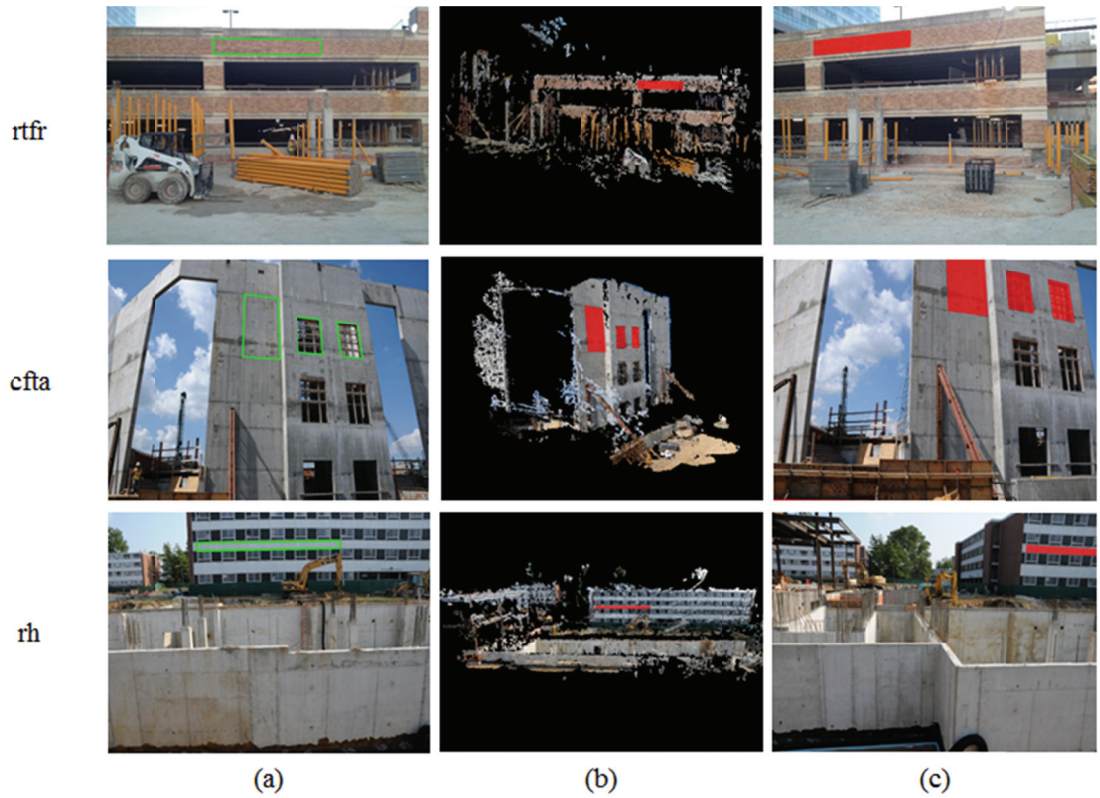

Figure 5.6 Results of 3D cyber-physical content authoring with the proposed method on street-scale outdoor data sets. (a) user-created information on the 2D image, (b) 3D elements driven from the user-created 2D elements, and (c) augmentation results of the user-created 3D cyber-information on another smart device on the site

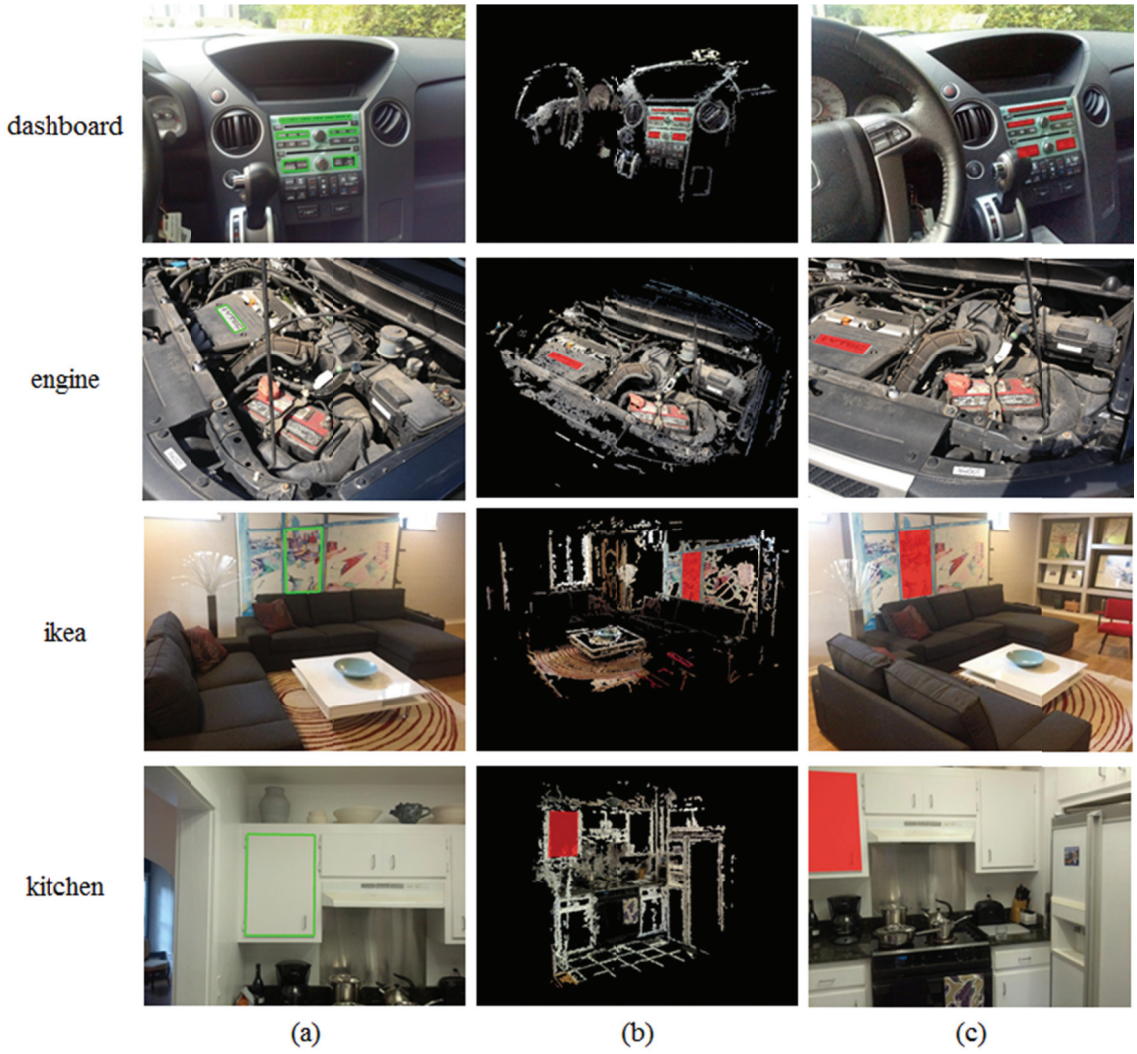

Figure 5.7 Results of 3D cyber-physical content authoring with the proposed method on room-scale indoor data sets. (a) user-created information on the 2D image, (b) 3D elements driven from the user-created 2D elements, and (c) augmentation results of the user-created 3D cyber-information on another smart device on the site

From experimental results shown in this section, we can conclude that the proposed method successfully creates 3D cyber-information solely based on user inputs on a single 2D image. By using a plane transformation, i.e., a homography matrix, to automatically find correspondences of user-created elements and triangulating all of those 2D correspondences using the recovered camera parameters, the proposed method automatically associates user-created cyber-information with corresponding physical objects in 3D geometry. As a result, users do not require manual association and a priori knowledge of the coordinates of underlying 3D physical model to create 3D cyber-information.

5.4 Contributions and Significance

Based on discussion in this Chapter, we can conclude that the solution approach, "Plane Transformation based 3D Cyber-physical Content Authoring from A Single 2D Image", has successfully filled the "Research Gap 2: 3D Cyber-physical Content Authoring from 2D Interface". Table 5.2 compares the proposed approach with all related works reviewed in Section 3.2. The plane transformation based 3D content authoring purely creates 3D cyber-information using user inputs from a single 2D image and supports automatic association of generated cyber-contents (e.g., product manual, history, website) to real-world 3D physical objects. In addition, the proposed method can be used with any commodity mobile devices if the devices have a capability of showing an image on the screen. The interface of the proposed method only requires a capability of drawing polygons on the image, and thus is intuitive and straightforward. The convenient method for 3D cyber-physical content authoring is especially important for designing and developing mobile augmented reality

applications where users can create and share cyber-information with each other in form of augmented reality overlays.

Table 5.2 Validation of the proposed approach – plane transformation based 3D cyber-physical content authoring from a single 2D image

Metrics	**3D drawings**	**Gesture recognition**	**Plane transformation based 3D content authoring**
External 3D framework	CAD	Not needed	Not needed
Automatic association with real-world objects	×	×	✓
Supports mobility	×	✓	✓
Device type	Personal Computer	Gloves, pens	Commodity smartphones

6 Cached *k-d* tree Generation for Fast Direct 2D-to-3D Matching

6.1 Overview of Solution Approach to Research Gap 3

As discussed in Chapter 4, the proposed HD^{4}AR approach, a vision-based marker-less method using SfM-based 3D physical models, provides near real-time mobile augmented reality with millimeter-level of information association. The HD^{4}AR show the robustness of the proposed approach to dynamic changes of viewpoint and scale of objects. Despite the accuracy and near real-time performance of the HD^{4}AR, however, the localization speed needs to be further accelerated to provide better user experience. With binary descriptors, the HD^{4}AR still takes 0.5-3.0 *sec* to localize a single photograph.

To fill the "Research Gap 3: Near Real-time Cyber-physical Information Association at Dynamically Varying Environmental Scale", here a new approach for further accelerating the HD^{4}AR localization/augmentation speed is designed and developed. As described in Section 4.3, the HD^{4}AR augmentation process is simply done by projecting 3D vertex points of cyber-information into an image plane using recovered camera parameters. However, the localization process requires a set of resource-intensive algorithms, such as direct 2D-to-3D matching algorithm, which performance depends on the number of 3D points in the 3D physical model. As a consequence, the longer localization time typically takes place at the outdoor data sets since the resulting 3D physical models are dense due to

a plenty of textures from the objects.

The matching complexity of the direct 2D-to-3D matching with a *k-d* tree proposed in Section 4.3 depends on the number of 3D points and the number of feature descriptors from a new image to be localized. Specifically, the upper bound of this matching complexity is:

$$O(M \log N) \tag{6.1}$$

where N is the number of 3D points in the point cloud and M is the number of feature descriptors from a new image. For outdoor data sets we studied in Section 4.4, the value of N is typically in the range between 30,000 and 200,000, while the value of M is 10,000-20,000. As shown in Equation 6.1, the larger N obviously results the longer matching time. If users create a 3D physical model of street or city using several hundreds of pre-collected photographs, the resulting model will consist of hundreds of thousands 3D points, and thus, a direct 2D-to-3D matching algorithm may take tens of seconds. Therefore, the methods of reducing the complexity of this direct 2D-to-3D matching are designed and proposed.

6.2 Caching 3D Representative Descriptors with Localization Patterns

Removing the dependency on the number of 3D points in Equation 6.1 can be expected to significantly reduce the overall matching time. To realize this, we developed a new approach that generates a constant size of cached *k-d* tree from 3D representative descriptors and using it for direct 2D-to-3D matching. By caching and maintaining highly

queried 3D points into a small size of *k-d* tree, the matching time and localization time are expected to be reduced.

With the proposed caching approach, a key question then becomes how to select which 3D points and their corresponding representative descriptors should be located in a cached *k-d* tree to provide high localization success-ratio and accurate localization results. To provide fast and reliable localization results, therefore, the proposed approach exploits the facts that 1) the HD^{4}AR accurately and rapidly localizes a new photograph with small number of 2D-to-3D correspondences and 2) localization requests from users may have a geospatial pattern, e.g., taking a picture only at façade of building. As a consequence, the most frequently matched 3D points during the previous localizations and their corresponding 3D representative descriptors are cached and used for fast direct 2D-to-3D matching.

The procedure of caching 3D points and corresponding representative descriptors can be summarized as follows:

1) After the 3D reconstruction process of the HD^{4}AR, create a "cache" list which size is equal to the number of 3D points in the 3D physical model. Each element of the list consists of (*hit count, Index of 3D point*) pair. The list will be maintained during an entire AR cycle of the HD^{4}AR.
2) After the direct 2D-to-3D matching stage in the HD^{4}AR localization, increase the hit count by 1 for all 3D points which exist in resulting 2D-to-3D correspondences.

3) Sort the "cache" list in decreasing order. The upper part of the list is the most frequently matched 3D points.
4) Extract 3D points and their corresponding representative descriptors according to the point indices of first *N* elements of the "cache" list. Typically the range of *N* is 1,000-10,000, depending on the size of the 3D physical model.
5) Generate a cached *k-d* tree using extracted 3D representative descriptors and use it for fast direct 2D-to-3D matching.

The localization process of the HD^{4}AR is slightly modified to handle fast direct 2D-to-3D matching with a cached *k-d* tree. Upon receiving a new photograph from the client device, the HD^{4}AR server first matches image feature descriptors of the new photograph against a cached *k-d* tree to find 2D-to-3D correspondences. If the number of correspondences is less than 16 or the HD^{4}AR was unable to calibrate the camera with resulting correspondences, the HD^{4}AR runs normal model-based 6-DOF localization discussed in Section 4.3 as a fallback solution. After the localization process, the HD^{4}AR updates the "cache" list and re-generates a cached *k-d* tree using updated information.

With a cached *k-d* tree, the complexity of direct 2D-to-3D matching is reduced to:

$$\mathrm{O}\,(M \log N) \;\rightarrow\; \mathrm{O}\,(M) \tag{6.1}$$

as *N* goes to constant. Since *M* is the number of feature descriptors of new photograph to be

localized and is completely a random number, it is difficult to remove the dependency of matching algorithm on *M*. However, by creating and using a constant size of 3D points, the proposed approach is believed to produce a similar level of matching times regardless of number of 3D points in the 3D physical model.

Figure 6.1 visualizes an example of cached 3D points after 25 random localization requests from client devices. The size of cache was set to 5,000 points so that the number of nodes in a cached *k-d* tree could not exceed 5,000. From Figure 6.1b, we can infer that the user localization requests mostly took place at the one side of the building in this test scenario and indeed had a geospatial pattern. By utilizing a cached *k-d* tree, the performance of the HD^{4}AR localization is up to 262% faster than the results provided in Chapter 4. The details of experimental results for the proposed approach will be fully discussed in Section 6.3.

Figure 6.1 An example of cached 3D physical model, (a) original 3D physical model, (b) caching the most frequently matched 3D points during the 25 localization requests. The size of cache is fixed to 5,000 points

6.3 Experimental Results and Validation

This section presents experimental results and the validation of the proposed caching approach for fast model-based localization using direct 2D-to-3D matching. In order to assess improvements provided by the proposed approach, the HD^{4}AR model-based 6-DOF localization discussed in Section 4.3 was performed on both cached models and non-cached models. In addition, only outdoor models were considered during this experiment as the outdoor models typically have larger number of 3D points and take longer localization time (2-3 *sec*) compared to indoor models. All 3D physical models used in this experiment, i.e., "patton", "knu", and "parliament" models, came from the results discussed in Section 4.4 and the details of the used physical models are reviewed in Table 6.1. In order to minimize feature extraction time during the localization, the BRISK descriptor is used in this experiment. The same photographs used in Section 4.4.2 were tested again for the proposed caching approach and the same metrics presented in Tables 4.12-4.14 were measured for performance comparison. Also, a half of test photographs were randomly selected to pre-train the "cache" list discussed in Section 6.2. All experiments were conducted on a single Amazon EC2 instance server with 22.5 GB memory and two Intel Xeon X5570 processors running Ubuntu version 12.04. An NVIDIA Tesla M2050 graphic card was used for GPU computations. The fallback solution – returning to normal model-based localization when the proposed caching approach failed to localize the photograph – was disabled during the experiment to assess the effect of the cache size on the localization success ratio.

During the experiment, different cache sizes, i.e., 1,000, 2,000, 5,000, and 10,000 points,

Table 6.1 3D physical models tested for direct 2D-to-3D matching with a cached *k-d* tree approach

Environment	Model Name	Number of 3D points	Mean re-projection error from 3D reconstruction
Outdoor	patton	46,318	0.498 pixels
	knu	33,122	0.552 pixels
	parliament	234,343	0.606 pixels

were mainly tested to validate the effect of the cache size on the performance. As discussed in Section 4.4.2, the average number of 2D-to-3D matches on outdoor building-scale data sets with BRISK descriptor was 1,271 points, meaning that the HD^{4}AR found about 1,000 points on average and used those 2D-to-3D correspondences to successfully recover the camera's location and orientation. Consequently, we can expect that the very small cache sizes, i.e., below 1,000 points, will achieve very low localization success-ratio. Tables 6.2-6.4 summarize the localization results of the proposed caching approach with very small cache sizes, i.e., 100- 500 points. As expected, the proposed approach with small cache size achieved very low localization success-ratio, which was in the range of 2-65%. As shown in Tables 6.2-6.4, the localization success-ratio depends on the gap between the average number of 2D-to-3D matches of non-caching localization and tested cache sizes. Specifically, the localization success-ratio significantly dropped when the gap between the average number of 2D-to-3D matches of non-caching localization and cache size was large. In the remaining of this section, therefore, we have mainly focused on cache sizes above 1,000 points to validate the effect of cache size on both localization speed and accuracy.

Table 6.2 Localization results with very small cache sizes for “patton” model

Package	**HD^{4}AR**	**HD^{4}AR with caching approach**		
Cache size	**-**	**100**	**200**	**500**
Localization success-ratio	49 / 50 (98%)	12 / 50 (24%)	12 / 50 (24%)	13 / 50 (26%)
Mean number of 2D-to-3D matches	2,145	50	94	191
Mean re-projection error	0.812 pixels	0.716 pixels	0.979 pixels	0.975 pixels
Mean localization time (sequential requests)	2.312 *sec* (1×)	1.241 *sec* (1.863×)	1.246 *sec* (1.856×)	1.254 *sec* (1.844×)
Mean localization time (parallel requests)	0.754 *sec* (1×)	0.432 *sec* (1.745×)	0.435 *sec* (1.733×)	0.442 *sec* (1.706×)

Table 6.3 Localization results with very small cache sizes for “knu” model

Package	**HD^{4}AR**	**HD^{4}AR with caching approach**		
Cache size	**-**	**100**	**200**	**500**
Localization success-ratio	50 / 50 (100%)	1 / 50 (2%)	26 / 50 (52%)	31/ 50 (62%)
Mean number of 2D-to-3D matches	1,204	8	20	41
Mean re-projection error	1.070 pixels	0.770 pixels	1.326 pixels	1.334 pixels
Mean localization time (sequential requests)	1.347 *sec* (1×)	0.697 *sec* (1.933×)	0.747 *sec* (1.803×)	0.781 *sec* (1.725×)
Mean localization time (parallel requests)	0.507 *sec* (1×)	0.304 *sec* (1.668×)	0.334 *sec* (1.518×)	0.347 *sec* (1.461×)

Table 6.4 Localization results with very small cache sizes for “parliament” model

Package	**HD^{4}AR**	**HD^{4}AR with caching approach**		
Cache size	**-**	**100**	**200**	**500**
Localization success-ratio	40 / 40 (100%)	16 / 40 (40%)	21 / 40 (52.5%)	26/ 40 (65%)
Mean number of 2D-to-3D matches	465	32	49	80
Mean re-projection error	0.897 pixels	0.654 pixels	0.743 pixels	0.793 pixels
Mean localization time (sequential requests)	2.693 *sec* (×)	0.974 *sec* (2.765×)	0.979 *sec* (2.751×)	0.998 *sec* (2.698×)
Mean localization time (parallel requests)	0.847 *sec* (1×)	0.284 *sec* (2.982×)	0.292 *sec* (2.901×)	0.300 *sec* (2.823×)

Table 6.5 compares the detail results of the caching approach on “patton” model which number of 3D points is 46,318 points. As shown in Table 6.5, the proposed caching approach achieved the fastest localization with the smallest cache size, while mean re-projection error remained the similar level to that of localizations without cache. However, the localization success-ratio with the small size of cache, i.e., 1,000-2,000 points, was slightly decreased compared to non-cache localization. This is due to the fact that a pre-trained cache does not properly cover the entire target scene as we selected the random photographs for caching 3D points. Nevertheless, the caching approach achieved 80-98% of localization success ratio and was 118-126% faster than the non-cache localization in all cases. To further demonstrate the acceleration factor of the proposed approach on direct 2D-to-3D matching, we also measured elapsed times for each step in localization, i.e., feature extraction time, and the matching/calibration time. As shown in Table 6.6, the matching and

calibration speed is improved by the caching approach, while the feature extraction time remains constant. Therefore, we can conclude that the proposed approach, which uses a cached *k-d* tree for matching, reduces overall localization time by reducing search space of direct 2D-to-3D matching. If we only consider the direct 2D-to-3D matching procedure, the

Table 6.5 Performance comparison of model-based 6-DOF localization for "patton" model

Package	**HD^{4}AR**	**HD^{4}AR with caching approach**			
Cache size	**-**	**1,000**	**2,000**	**5,000**	**10,000**
Localization success-ratio	49 / 50 (98%)	40 / 50 (80%)	44 / 50 (88%)	49 / 50 (98%)	49 / 50 (98%)
Mean number of 2D-to-3D matches	2,145	134	228	438	748
Mean re-projection error	0.812 pixels	0.962 pixels	0.927 pixels	1.047 pixels	1.060 pixels
Mean localization time (sequential requests)	2.312 *sec* (1×)	1.314 *sec* (1.760×)	1.484 *sec* (1.558×)	1.692 *sec* (1.366×)	1.836 *sec* (1.259×)
Mean localization time (parallel requests)	0.754 *sec* (1×)	0.477 *sec* (1.581×)	0.547 *sec* (1.378×)	0.583 *sec* (1.378×)	0.627 *sec* (1.203×)

Table 6.6 Details of localization time for sequential requests on "patton" model

Package	**HD^{4}AR**	**HD^{4}AR with caching approach**			
Cache size	**-**	**1,000**	**2,000**	**5,000**	**10,000**
BRISK feature extraction time	0.785 *sec*	0.785 *sec*	0.785 *sec*	0.785 *sec*	0.785 *sec*
Matching/ calibration time (performance gain)	1.527 *sec* (1×)	0.529 *sec* (2.887×)	0.698 *sec* (2.188×)	0.907 *sec* (1.684×)	1.050 *sec* (1.454×)

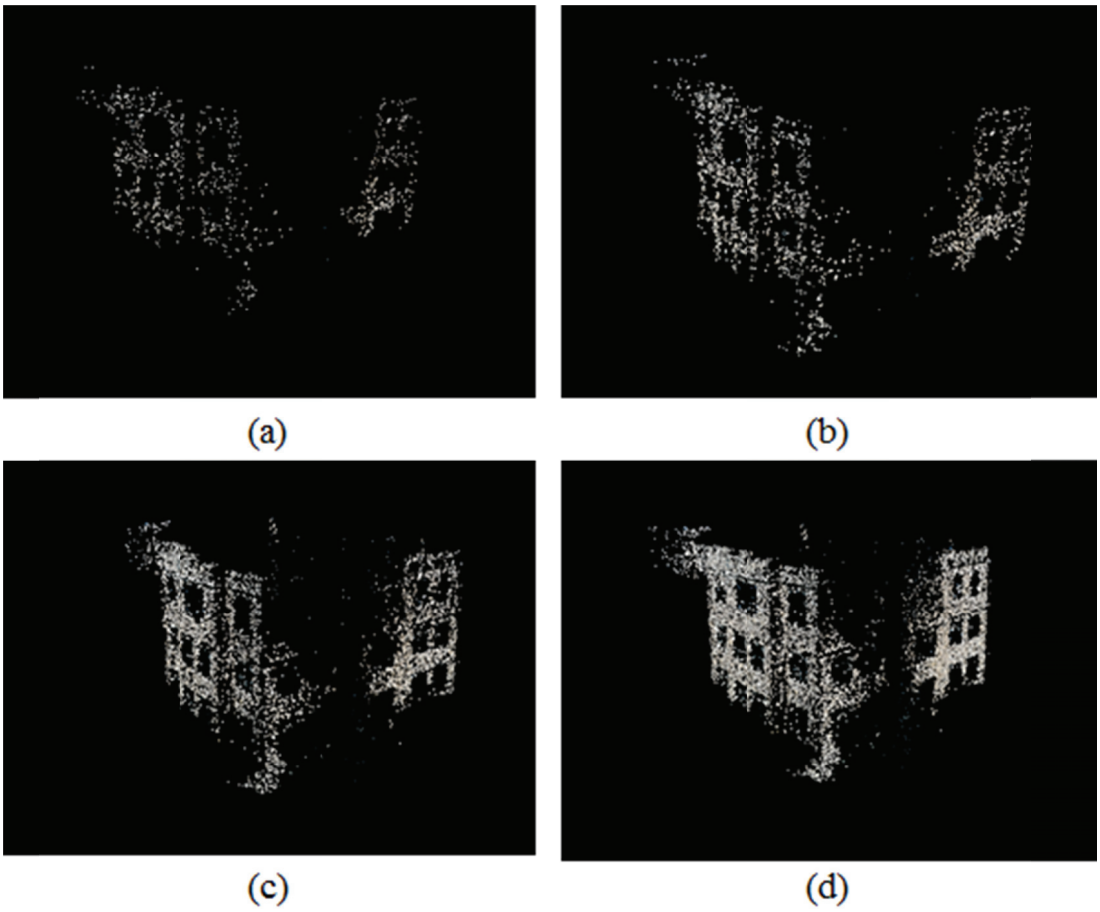

Figure 6.2 Cached 3D physical models of the "patton" model, (a) cache size = 1,000 points, (b) cache size = 2,000 points, (c) cache size = 5,000 points, and (d) cache size = 10,000 points

matching/calibration time was up to 2.887 times faster than the non-cache localization discussed in Section 4.3. Figure 6.2 visualizes the cached 3D physical models with different cache sizes. As expected, the smaller cache sizes produced more spare 3D point clouds, but the proposed approach successfully localized most of photographs even with these sparse cached 3D point clouds.

Tables 6.7-6.8 compare the detail results of the caching approach on “knu” model, which number of 3D points is 33,122 points. Again, the proposed caching approach achieved the fastest localization with the smallest cache size, while mean re-projection error was slightly increased. For “knu” model, however, the localization success-ratio was not decreased even with small cache sizes. As shown in Figure 6.3, the cached 3D models were well-trained and covered the entire target scene even when cache size was 1,000 points. The performance gain of the caching approach is 118-158% on localization and 131-226% on direct 2D-to-3D matching. As the “knu” model has less number of 3D points than “patton” model, the performance gain is slightly decreased. However, the proposed approach was faster than the non-cache localization and achieved the overall localization time under 1 sec for “knu” model.

Table 6.7 Performance comparison of model-based 6-DOF localization for “knu” model

Package	**HD^{4}AR**	**HD^{4}AR with caching approach**			
Cache size	**-**	**1,000**	**2,000**	**5,000**	**10,000**
Localization success-ratio	50 / 50 (100%)	49 / 50 (98%)	50 / 50 (100%)	50 / 50 (100%)	50 / 50 (100%)
Mean number of 2D-to-3D matches	1,204	87	157	338	561
Mean re-projection error	1.070 pixels	1.457 pixels	1.504 pixels	1.536 pixels	1.396 pixels
Mean localization time (sequential requests)	1.347 *sec* (1×)	0.854 *sec* (1.577×)	0.959 *sec* (1.405×)	1.033 *sec* (1.304×)	1.138 *sec* (1.184×)
Mean localization time (parallel requests)	0.507 *sec* (1×)	0.386 *sec* (1.313×)	0.414 *sec* (1.225×)	0.440 *sec* (1.152×)	0.470 *sec* (1.079×)

Table 6.8 Details of localization time for sequential requests on “knu” model

Package	**HD^{4}AR**	**HD^{4}AR with caching approach**			
Cache size	**-**	**1,000**	**2,000**	**5,000**	**10,000**
BRISK feature extraction time	0.462 *sec*	0.462 *sec*	0.462 *sec*	0.462 *sec*	0.462 *sec*
Matching/ calibration time (performance gain)	0.886 *sec* (1×)	0.392 *sec* (2.260×)	0.497 *sec* (1.783×)	0.572 *sec* (1.549×)	0.677 *sec* (1.309×)

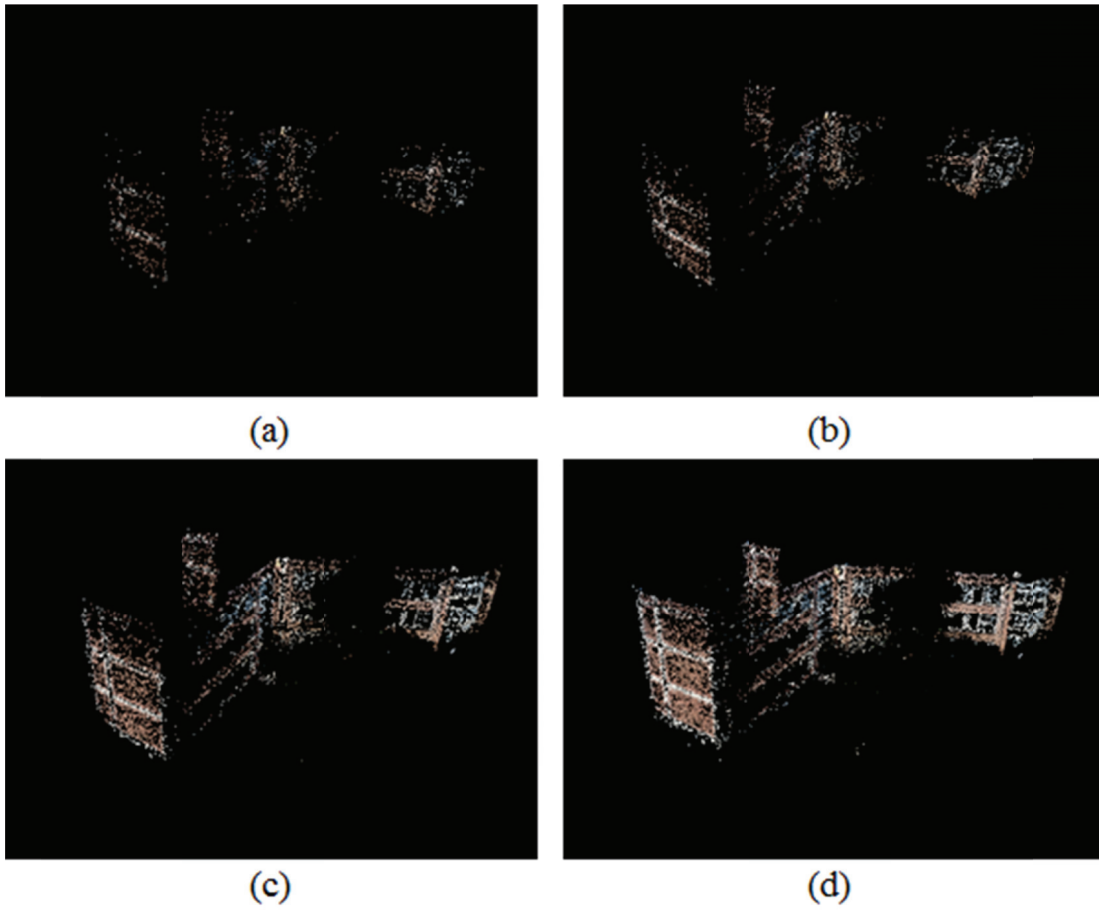

Figure 6.3 Cached 3D physical models of the “knu” model, (a) cache size = 1,000 points, (b) cache size = 2,000 points, (c) cache size = 5,000 points, and (d) cache size = 10,000 points

Finally, the proposed caching approach was applied to a large-scale model, i.e., "parliament" model. The number of 3D points in "parliament" model is 234,343 points. Tables 6.9-6.10 compare the results of the caching approach on "parliament" model and Figure 6.4 presents the cached 3D physical models with different cache sizes. As shown in Tables 6.9-6.10, the localization with a cache significantly improved the localization speed and matching speed for "parliament" model. The proposed approach was 196-262% faster than the non-cache localization and the direct 2D-to-3D matching was up to 465% faster. In addition, the mean re-projection error remained the similar level to that of non-cache localization even with cache size of 1,000 points. From these results, we can conclude that the proposed caching approach has improved the performance of model-based 6-DOFlocalization on large-scale physical models and provides reliable and accurate localization results.

Table 6.9 Performance comparison of model-based 6-DOF localization for "parliament" model

Package	**HD^{4}AR**	**HD^{4}AR with caching approach**			
Cache size	**-**	**1,000**	**2,000**	**5,000**	**10,000**
Localization success-ratio	40 / 40 (100%)	37 / 40 (92.5%)	37 / 40 (92.5%)	40 / 40 (100%)	40 / 40 (100%)
Mean number of 2D-to-3D matches	465	104	178	337	442
Mean re-projection error	0.897 pixels	0.990 pixels	0.906 pixels	0.858 pixels	0.872 pixels
Mean localization time (sequential requests)	2.693 *sec* (×)	1.027 *sec* (2.622×)	1.134 *sec* (2.375×)	1.301 *sec* (2.070×)	1.377 *sec* (1.956×)
Mean localization time (parallel requests)	0.847 *sec* (1×)	0.345 *sec* (2.455×)	0.369 *sec* (2.295×)	0.415 *sec* (2.041×)	0.439 *sec* (1929×)

Table 6.10 Details of localization time for sequential requests on “parliament” model

Package	**HD^{4}AR**	**HD^{4}AR with caching approach**			
Cache size	**-**	**1,000**	**2,000**	**5,000**	**10,000**
BRISK feature extraction time	0.571 *sec*	0.571 *sec*	0.571 *sec*	0.571 *sec*	0.571 *sec*
Matching/ calibration time (performance gain)	2.122 *sec* (1×)	0.456 *sec* (4.654×)	0.563 *sec* (3.769×)	0.730 *sec* (2.907×)	0.806 *sec* (2.633×)

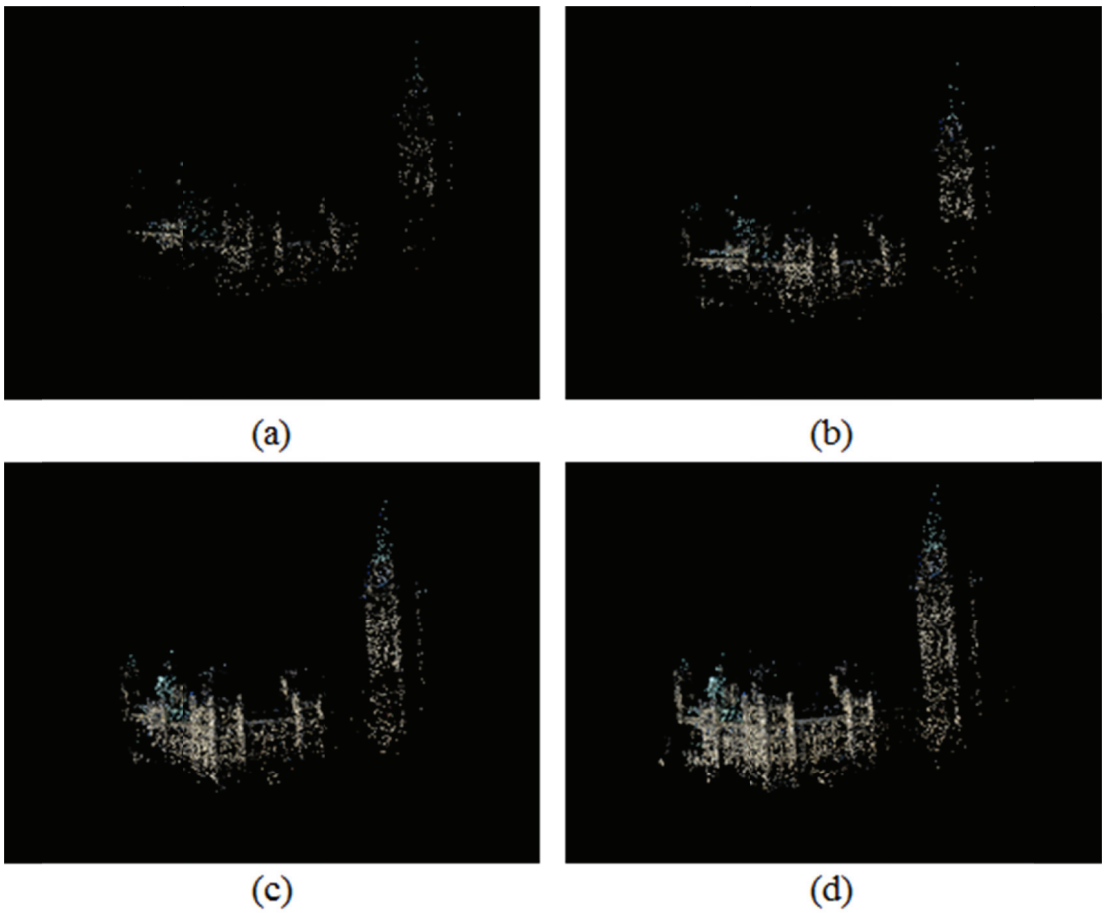

Figure 6.4 Cached 3D physical models of the “parliament” model, (a) cache size = 1,000 points, (b) cache size = 2,000 points, (c) cache size = 5,000 points, and (d) cache size = 10,000 points

6.4 Contributions and Significance

Based on discussion in this Chapter, we can conclude that the solution approach, i.e., "Cached *k-d* tree generation for Fast Direct 2D-to-3D matching", is a novel approach that brings caching scheme into a direct 2D-to-3D matching algorithm used in model-based localization. No existing work to date attempts to improve the speed of model-based localization by tackling the complexity of direct 2D-to-3D matching. By removing the dependency on number of 3D points, the proposed approach provides near real-time localization/augmentation results regardless of number of 3D points in the 3D physical model. Table 6.11 summarizes the proposed approach with all related works reviewed in Section 3.3. With the proposed approach, the localization time now takes at most 1.5 *sec* for

Table 6.11 Validation of the proposed approach – cached k-d tree generation for fast direct 2D-to-3D matching

Metrics	**Model-based**	**HD^{4}AR with caching approach**
Model scale	room-street	object-street
Model preparation time	3 – 24 *hr*	0.1 – 1 *hr*
Number of 3D physical models in the system	Single	Single[(a)]
Number of cyber-information items in the system	$0 – 10^3$	$10^0 – 10^4$
Localization/ Augmentation Speed	5 – 240 *sec*	0.5 – 1 *sec*
Supports mobility	×	✓

[(a)] The scenario for multiple models will be discussed in Chapter 7

large-scale physical models. In addition, it still achieves the high-precision localization with maximum error of few image pixels. Therefore, the proposed caching approach for fast localization successfully fills the “Research Gap 3: Near Real-time Cyber-Physical Information Association at Dynamically Varying Environmental Scales”.

7 Multi-model based 6-DOF Localization for Blinded Localization Requests

7.1 Overview of Solution Approach to Research Gap 3

To fill the "Research Gap 3: Near Real-time Cyber-physical Information Association at Dynamically Varying Environmental Scale", new solution approaches for large-scale model-based 6-DOF localization are developed and validated in this chapter.

All solution approaches presented in Chapters 4-6 assume that there is only a single 3D physical model in the system or users know which model should be used for localization and augmentation. For example, let us assume that separate point cloud models were created for different locations/objects in the HD^{4}AR server. Then, users are required to choose the model from a list on the client device and enable model-based localization with respect to the corresponding 3D physical model. This strategy is impractical when the number of physical models is enormous and/or users do not know which model should be used for localization and augmentation. To overcome this issue and provide near real-time localization/augmentation service in the presence of multiple 3D physical models, we develop a new approach which can handle the localization requests that do not know the target physical model for localization. Throughout this chapter, we will refer the localization request that do not indicate the target 3D physical model as *blinded* localization request.

7.2 Double-stage Matching Algorithm with A Single Indexed *k-d* tree

The straightforward way for finding an appropriate model for blinded localization is matching a new image from users to all 3D physical models in the server sequentially. Then, the localization is done when the certain 3D model successfully localizes a given photograph. Obviously, this sequential matching is very time-consuming and is inefficient if the target model exists at the end of the model list. Specifically, the upper bound of this sequential matching complexity is:

$$O(K M \log N) \tag{7.1}$$

where K is the number of models that exist in the server, N is the number of 3D points in each physical model, and M is the number of feature descriptors from a new image to be localized. For outdoor data sets we studied in Section 4.4, the value of N is typically in the range between 30,000 and 200,000 while the value of M is 10,000-20,000.

Instead of time-consuming sequential matching, we propose to create a single indexed *k-d* tree and use it to find the target model for blinded localization requests. Specifically, a single *k-d* tree is created by concatenating all 3D representative descriptors from multiple models, and model index information is imposed to each 3D representative descriptor. After matching a new image against this single indexed *k-d* tree, the 3D physical model that has the largest number of 2D-to-3D matches will be the model to be used for localization and augmentation. Then, the model-based 6-DOF localization or caching approach discussed in

Chapters 4 and 6 can be used to localize a given photograph. The procedure of this double-stage matching algorithm with a single indexed *k-d* tree can be summarized as follows:

1) Concatenate all 3D representative descriptors from 3D physical models presented in the HD^{4}AR server. Also, the model index information is imposed to each 3D representative descriptor to indicate which model has the corresponding descriptor.
2) Upon receiving a blinded localization request from the client, perform the direct 2D-to-3D matching between given image and a generated indexed *k-d* tree.
3) By using found 2D-to-3D correspondences and the model index information, count the number of 2D-to-3D matches for each 3D physical model.
4) Take *N* models that have the largest number of matches. Then, perform the model-based 6-DOF localization for each model in parallel. The value of *N* is typically set to 1-3.
5) Select the localization result which has the minimum re-projection error and return it to the client.

The proposed double-stage matching algorithm can be reduced to a single-stage matching as the result of first-stage matching already includes the 2D-to-3D correspondences of the target model. However, the reason of double-stage matching is for the case that several models have very similar visual features and thus are not clearly distinguished from each other through first-stage matching. For example, if two 3D physical models *A* and *B* are created for the same building, it is possible that some of 2D-to-3D correspondences found

in the first-stage matching belong to model *A* while some of 2D-to-3D correspondences belong to model *B*. As discussed in Chapters 4 and 6, the less number of 2D-to-3D correspondences decreases the accuracy of localization. Therefore, we utilize the first-stage matching results only for finding candidate target models and perform the second-stage matching in parallel to get the most accurate localization results.

With the proposed approach, the complexity of blinded localization is reduced to:

$$\mathrm{O}\,(K\,M\log N)\;\rightarrow\;\mathrm{O}\,(M\log K+2M\log N) \tag{7.2}$$

where K is the number of models that exist in the server, N is the number of 3D points in each physical model, and M is the number of feature descriptors from a new image. The details of the performance gain provided by the proposed single indexed *k-d* tree approach will be fully discussed in Section 7.4.

7.3 K-means Clustering of 3D Physical Models with Geo-information

Another way of finding a target model for blinded localization requests is exploiting the geo-information which can be easily obtained by modern commodity smartphones. This approach segments large-scale 3D physical models into several clusters and automatically finds an appropriate cluster to localize and augment a new photograph sent from the client device. To cluster 3D physical models, we use GPS latitude and longitude values measured by mobile device and recorded in the image in form of EXIF (Exchangeable Image File

Format) tag. There is no need for accurate GPS values as we only use this information for clustering purposes. The overall steps for clustering a 3D physical model are:

1) *Partitioning base images*: All base images participated in 3D reconstruction are divided into several clusters using latitude and longitude values of each base image. In order to find the proper number of clusters, hierarchical clustering analysis [54] is first used to estimate starting values for the K-means algorithm [55]. Based on the resulting number of clusters, the K-means algorithm is performed to partition base images to each cluster with the nearest mean of GPS values. Specifically, K-means algorithm partitions n base images into k clusters that each base image belongs to each cluster with the nearest mean:

$$\arg\min \sum_{i=1}^{k} \sum_{X_j \in S_i} ||X_j - \mu_i||^2 \tag{7.3}$$

where k is the number clusters, μ_i is the mean of GPS values for S_i cluster. After computing center of each cluster and all images are assigned to each closet clusters, the cluster centers are recomputed based on the mean values of all GPS values in the cluster. This procedure is done iteratively until the variance of each cluster is small enough [56].

2) *Clustering a 3D physical model*: Once the base images are successfully partitioned, we segment the 3D point clouds by selecting 3D points and their corresponding

representative descriptors that are observed by base images in each cluster. As a consequence, each clustered point clouds contains less 3D points compared to initial 3D physical models, resulting smaller scale.

The localization process is slightly modified to handle clustered 3D physical models. With the proposed approach, upon receiving a new photograph from the client device, the HD^{4}AR server first finds the nearest cluster by comparing GPS values recorded in the new photograph to mean value of each cluster. After finding the nearest cluster, the server performs existing model-based 6-DOF localization method discussed in Chapters 4 and 6 to compute a complete pose of the camera. If the new photograph does not include GPS tag, the server attempts to localize the image with all clusters in parallel. Although the proposed approach requires mobile devices to enable GPS sensors during the AR cycle, the clustering approach can handle blinded localization requests by reading a GPS value recorded in the image and also result faster localization due to smaller scale of 3D point clouds. The details of experimental results will be fully discussed in following section.

7.4 Experimental Results and Validation

This section presents experimental results and the validation of the proposed solution approaches, i.e., “Double-stage Matching Algorithm with A Single Indexed *k-d* tree” and “K-means Clustering of 3D Physical Models with Geo-information”. Therefore, two separate experiments, i.e., Multiple-model based localization and Localization with Clustered 3D Physical Models, were performed and validated. The details of the data set

specifications and validation metrics are discussed in the following subsections. After showing experimental results, the overall validation of solution approaches will be summarized.

7.4.1 Multiple-model Based Localization

The multiple-model based localization was first tested with the proposed double-stage matching algorithm using a single indexed *k-d* tree. To emulate an environment where multiple 3D physical models exist in the server, we used total 200 physical models generated from the 3D reconstruction process discussed in Section 4.2. Among 200 physical models, the 10 models came from the results presented in Section 4.4.1. The details of test scenarios are summarized in Table 7.1. The server side of the HD^{4}AR for localization was running on Ubuntu version 12.04 with 8 GB memory and a 4-core Intel i5-2520M processor. Also, the BRISK descriptor is used for this experiment.

Table 7.1 3D physical model specifications for multi-model based localization experiment

Number of 3D models	Total number of 3D points	Total point cloud size
10	484,006	201.21 MB
20	1,238,784	503.33 MB
60	2,647,207	1.07 GB
100	3,374,138	1.38 GB
200	4,095,305	1.70 GB

In order to validate that the proposed approach can successfully find target models for blinded localization requests, a group of successfully localized photographs from Section 4.4.2 were tested without designating the target models. In addition, only the performance of sequential localizations from a single client device was measured. Tables 7.2 shows the overall results of the proposed double-stage matching approach for multi-model based localizations. As shown in Table 7.2, the proposed double-stage matching algorithm with a single indexed *k-d* tree approach successfully found target models for all blinded localization requests regardless of the number of models in the system. In addition, the proposed approach rapidly and accurately localized all tested photographs even in the presence of 200 models in the system. The mean localization times for multi-model based localizations were in the range between 1.360-2.623 *sec* and the mean re-projection errors were within 1.507-1.532 pixels.

Table 7.2 Performance comparison of multi-model based localization

Number of models in the system	**10**	**20**	**60**	**100**	**200**
Localization success-ratio	235 / 235 (100%)	235 / 235 (100%)	235 / 235 (100%)	235 / 235 (100%)	235 / 235 (100%)
Mean number of 2D-to-3D matches	523	524	537	537	537
Mean re-projection error	1.531 pixels	1.532 pixels	1.513 pixels	1.511 pixels	1.507 pixels
Mean localization time	1.360 *sec*	1.568 *sec*	2.054 *sec*	2.343 *sec*	2.623 *sec*

To further demonstrate the performance factors of the proposed approach, we also measured the elapsed times for each step in localization, i.e., target model searching time, feature extraction time, and the matching/calibration time. As shown in Table 7.3, the target model searching time, which corresponds to the first-stage matching time in the proposed approach, only took 0.482-1.799 *sec* in our test scenarios where the number of models are varied from 10 to 200. As expected in Section 7.2, the target model searching time is not proportional to the number of models. Even in the presence of 200 models, the target model searching with the proposed approach took under 2 *sec*. From experimental results shown in this section, we can conclude that the proposed approach successfully handles the blinded localization requests and provides near real-time localization/augmentation in the presence of multiple 3D physical models in the system. In addition, the experimental results imply that the double-stage matching algorithm with a single indexed *k-d* tree approach can be extended to hundreds of 3D physical models without significantly reducing the localization performance.

Table 7.3 Details of localization time from the proposed single indexed *k-d* tree approach

Number of models in the system	**10**	**20**	**60**	**100**	**200**
Target model searching time	0.482 *sec*	0.738 *sec*	1.222 *sec*	1.509 *sec*	1.799 *sec*
BRISK feature extraction time	0.570 *sec*	0.573 *sec*	0.571 *sec*	0.578 *sec*	0.566 *sec*
Matching/ calibration time	0.308 *sec*	0.257 *sec*	0.261 *sec*	0.256 *sec*	0.258 *sec*

7.4.2 Localization with Clustered 3D Physical Models

To validate the clustering approach discussed in Section 7.3, we enabled the GPS sensor installed in smartphones and recorded its values in form of EXIF tag during a photo collection for 3D reconstruction. Then, the HD^{4}AR 3D reconstruction procedure discussed in Section 4.2 was performed on newly collected base images. During the 3D reconstruction, the FREAK descriptor is used to minimize feature extraction time and memory consumption. The resulting 3D physical model was then partitioned into three clusters using GPS values of each base image and K-means clustering algorithm. The final results of 3D reconstruction and clustering are summarized in Table 7.4. The results show that the 3D physical model was successfully reconstructed and well-partitioned into three clusters. Figure 7.1 visualizes the original 3D physical model and its corresponding clusters, showing that the original physical model was geologically partitioned.

Table 7.4 Results of 3D reconstruction and clustering

	Original 3D physical model	**Cluster #1**	**Cluster #2**	**Cluster #3**
Number of base images	66	15	21	30
Number of 3D points	70,906	24,178	23,098	27,528
Mean re-projection error	0.523 pixels	0.511 pixels	0.553 pixels	0.608 pixels
GPS mean value (latitude, longitude)	(37.2290, -80.4225)	(37.2293, -80.4227)	(37.2289, -80.4227)	(37.2290, -80.4222)

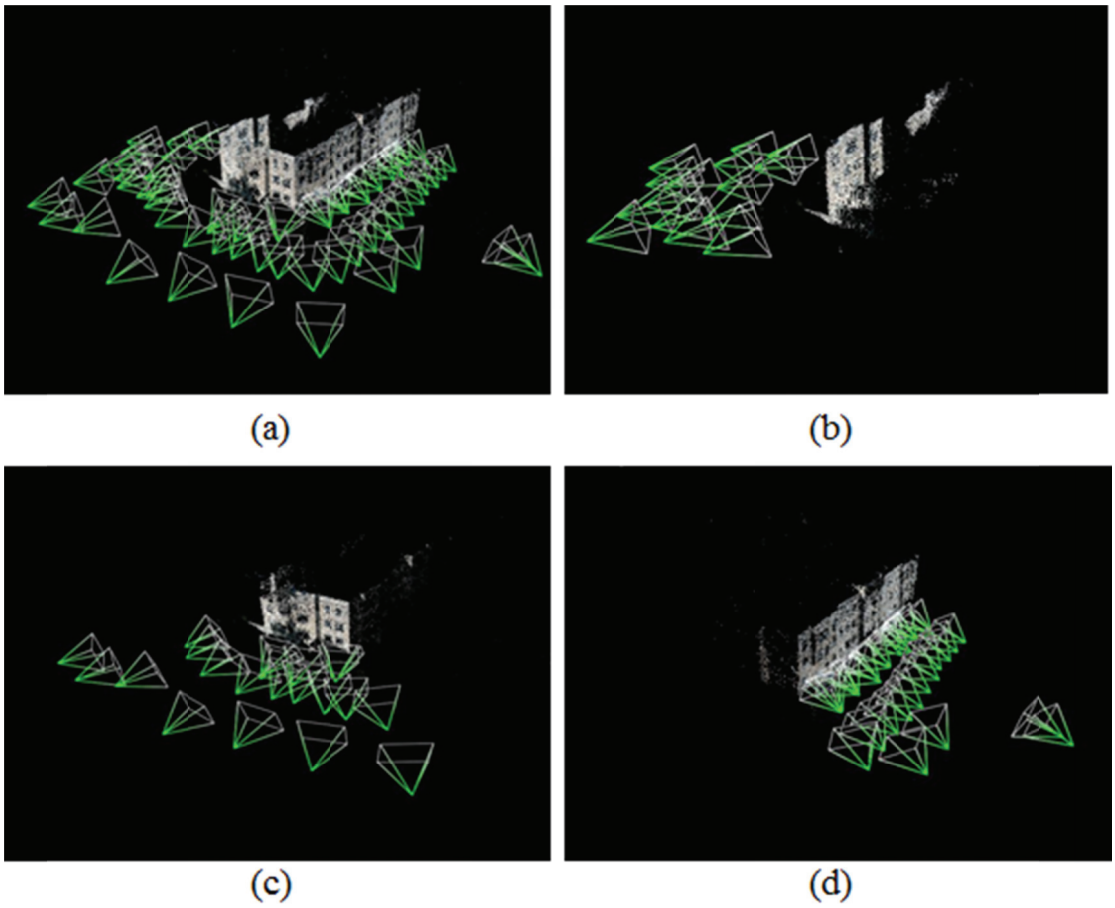

Figure 7.1 Resulting 3D point clouds with the HD^{4}AR and proposed clustering method; (a) Original 3D physical model, (b) cluster #1, (c) cluster #2, and (d) cluster #3

After clustering the 3D physical model, the localization success-ratio, mean re-projection error, and the elapsed time using clustered 3D physical model were measured and compared to results using non-clustered single physical model. In this experiment, we only measured the localization performance with the sequential requests from a single device although the HD^{4}AR can handle multiple requests of localization from several client devices simultaneously, which leads to increased system capacity. The server side of the HD^{4}AR

was running on Windows 7 with 8 GB memory and a 4-core Intel i7-870 processor. As observed in Table 7.5, the experimental results show that the clustering approach successfully found the target cluster by using geo-location data of the given photograph and resulted in 100% success-ratio of localization. The mean re-projection error of localized photographs with each cluster presents single-pixel error in all cases. In addition, the proposed approach accelerates the overall localization speed up to 154% with the tested data set, without reducing success-ratio and mean re-projection error.

To further demonstrate the acceleration factor of the proposed approach, we also measured elapsed times for each step in localization, i.e., cluster selection time, feature extraction time, and the matching/calibration time. As shown in Table 7.6, the matching and calibration took the longer time when the size of 3D physical model (i.e., number of 3D points) is larger, while the feature extraction time remained constant. Therefore, the proposed clustering approach, which segments the large-scale physical model into smaller

Table 7.5 Performance of model-based 6-DOF localization with clustered 3D physical models

	Original 3D physical model	**Cluster #1**	**Cluster #2**	**Cluster #3**
Localization success-ratio	75 / 75 (100%)	25 / 25 (100%)	25 / 25 (100%)	25 / 25 (100%)
Mean re-projection error	0.958 pixels	0.937 pixels	0.960 pixels	1.037 pixels
Mean localization time (performance gain)	2.735 *sec* (1×)	1.897 *sec* (1.442×)	1.782 *sec* (1.535×)	1.934 *sec* (1.414×)

Table 7.6 Details of the localization time with clustered 3D physical models

	Original 3D physical model	**Cluster #1**	**Cluster #2**	**Cluster #3**
Cluster selection time	0 *sec*	3.5×10^{-7} *sec*	3.5×10^{-7} *sec*	3.5×10^{-7} *sec*
BRISK feature extraction time	0.759 sec	0.775 sec	0.755 sec	0.760 sec
Matching/ calibration time (performance gain)	1.976 *sec* (1×)	1.122 *sec* (1.761×)	1.027 *sec* (1.924×)	1.174 *sec* (1.683×)

physical models, not only supports the blinded localization requests, but also reduces overall localization time by reducing the size of 3D physical model. If we only consider the direct 2D-to-3D matching procedure, the matching/calibration time was up to 1.924 times faster than the non-clustered model-based localization.

7.5 Contributions and Significance

Based on discussion in this Chapter, we can conclude that the solution approaches, i.e., "Double-stage Matching Algorithm with A Single Indexed *k-d* tree" and "K-means Clustering of 3D Physical Models with Geo-information" successfully fills the "Research Gap 3: Near Real-time Cyber-Physical Information Association at Dynamically Varying Environmental Scales". Specifically, the proposed solution approaches can provide near real-time, high-precision mobile augmented reality in the presence of hundreds of 3D physical models in the system. Table 7.7 summarizes the proposed approach with all related works reviewed in Section 3.3. All previous work related to model-based localization

methods only considers the case when there is a single model in the system.

As discussed in Section 7.4, the proposed double-stage matching algorithm using a single indexed *k-d* tree can rapidly find target models for blinded localization requests and successfully localize the photographs under 3 *sec* with 200 physical models in the system. In addition, a new clustering approach using geo-information is developed and validated to handle large-scale physical models and further accelerate the model-based localization speed. The large-scale physical models can be successfully partitioned into several clusters using the proposed approach and the blinded localization requests can always be matched against correct clusters by using geo-information obtained through the sensor installed in commodity mobile devices.

Table 7.7 Validation of the proposed approaches for multi-model based 6-DOF localization

Metrics	**Model-based**	**HD^{4}AR with solution approaches**
Model scale	room-street	object-street
Model preparation time	3 – 24 *hr*	0.1 – 1 *hr*
Number of 3D physical models in the system	Single	Multiple (Hundreds of models)
Number of cyber-information items in the system	$0 – 10^3$	$10^0 – 10^4$
Localization/ Augmentation Speed	5 – 240 *sec*	1 – 2 *sec*
Supports mobility	×	✓

8 Conclusions

This dissertation presents a new, fast, and scalable Structure-from-Motion (SfM) approach for high-precision mobile augmented reality systems. To develop solution approaches, current open research problems and research gaps in mobile augmented reality are first scrutinized. Based on our investigation provided in this dissertation, current research gaps in mobile augmented reality can be summarized as: 1) fine-grained 6-DOF localization with mobile devices, 2) 3D cyber-physical content authoring from 2D interface, and 3) near real-time cyber-physical information association at dynamically varying environmental scales.

To fill these research gaps, total five solution approaches are developed and validated: 1) Hybrid 4-Dimensional Augmented Reality (HD^4AR), 2) Plane transformation based 3D cyber-physical content authoring from a single 2D image, 3) Cached *k-d* tree generation for fast direct 2D-to-3D matching, 4) Double-stage matching algorithm with a single indexed *k-d* tree, and 5) K-means Clustering of 3D physical models with geo-information. In following sections, the contributions of each solution approach are summarized and the future work of this study is identified.

8.1 Summary of Contributions

Provide near real-time millimeter-accuracy overlay of cyber-information associated with real-world physical objects in 3D geometry using commodity mobile devices (completed

and presented in Chapter 4).

To fill the first research gap, a novel hybrid approach, which combines model-based 6-DOF localization and SfM-based model generation, is developed and validated. The overall approach is called Hybrid 4-Dimensional Augmented Reality (HD^4AR) which purely localizes users based on images from a mobile device and does not require any sensors or infrastructures for mobile augmented reality. By introducing a new parallelized SfM process, which accelerates an existing 3D reconstruction pipeline by a factor of 30, the HD^4AR makes model-based localization feasible in mobile augmented reality and provides much shorter model preparation time compared to existing work. In addition, the proposed model-based 6-DOF localization method using direct 2D-to-3D matching speeds up existing works by a factor of 160. The HD^4AR only takes 0.5-3.0 *sec* to localize a single photograph and the uncertainty level of localization is 0.613-2.511 pixels. Finally, experimental results show that the HD^4AR can provide millimeter-level information association accuracy in both indoor and outdoor environment, from room-level to street-level scales.

Enable 3D cyber-physical content authoring from limited 2D user interfaces (completed and presented in Chapter 5).

Along with the HD^4AR approach, a new plane transformation based 3D cyber-physical content authoring approach is proposed and validated to fill the second research gap. The proposed approach purely creates 3D cyber-information using user inputs on a single 2D image and automatically associates user-created cyber-information with corresponding

physical objects in 3D geometry. Validation results show that all user-created elements on 2D images can be accurately triangulated and associated with objects in 3D physical models, and the generated 3D cyber-information can be precisely overlaid on other photographs taken at completely different locations. By considering a fact that the 3D content authoring from 2D interface is still an open problem, the proposed approach can address the open research problem and make 3D cyber-physical content authoring feasible on any commodity mobile devices.

Provide a localization method which operates in near real-time at large-scale environment (completed and presented in Chapter 6).

Another solution approach, i.e., a cached *k-d* tree generation, is suggested and validated to further enhance the model-based localization speed with large-scale 3D physical models. By grafting caching scheme into direct 2D-to-3D matching algorithm, the matching complexity is significantly reduced. No existing work to date attempts to improve the speed of model-based localization by tackling the complexity of direct 2D-to-3D matching. By removing the dependency of direct 2D-to-3D matching on number of 3D points, the proposed approach provides near real-time localization/augmentation results regardless of number of 3D points in the 3D physical model. With the proposed approach, the localization time now takes at most 1 *sec* for large-scale physical models. In addition, it still achieves the high-precision localization with the maximum error of up to 1 image pixel.

Provide a fast cyber-physical information association method for multiple and combined physical scales (completed and presented in Chapters 7-8).

Finally, two solution approaches, i.e., double-stage matching algorithm with a single indexed *k-d* tree and K-means Clustering of 3D physical models with geo-information, are developed and validated to provide high-precision mobile augmented reality in the presence of multiple physical models in the system. The proposed double-stage matching algorithm can rapidly find the target models for blinded localization requests and successfully localize the photographs in near real-time even with hundreds of 3D physical models in the system. As a consequence, the mobile augmented reality systems can be easily extended to tons of users creating different 3D physical models separately, and the users do not require a priori knowledge of target model for multiple-model based 6-DOF localization. In addition, a new clustering approach using geo-information is developed to handle large-scale physical models and further accelerate the localization speed. The large-scale physical models can be successfully partitioned using the proposed approach and the blinded localization requests can always be matched against correct clusters by using geo-information obtained through the sensor installed in commodity mobile devices.

By combining all these proposed solution approaches, which simplify and speed up the process of accurately obtaining relevant cyber-information, the output of research can be used in many practical context-aware applications, such as construction progress monitoring or monitoring the manufacture of electronic circuit boards, etc. Since the proposed solution can work with commodity smartphones and does not depend on external

devices, such as GPS satellites, optical markers, or geomagnetic sensors, the application of the solution in the field is expected to be inexpensive and practical.

8.2 Future Work

While this study presents the promising results toward near real-time high-precision mobile augmented reality by developing hybrid mobile/cloud model-based localization on SfM-based 3D physical models, some research challenges needs to be addressed for better mobile augmented reality applications:

1) *Real-time localization/augmentation*: although the HD^{4}AR achieves near real-time localization regardless of environmental constraints, some applications, such as AR-based video gaming, may require real-time augmented reality. A possible solution is to develop a hybrid approach using both image and supplemental sensors installed in commodity smartphones. For example, key frames in the video are localized through model-based approach proposed in this study while intermediate frames are localized through inertial or geomagnetic sensors.
2) *Minimal number of base images*: we typically collected about 50-100 images for each target scene to produce 3D physical models. This number came from our empirical experiments, and therefore, the relationship between number of base images and the quality of 3D point cloud should be further analyzed to guide users to take a minimal number of base images for bootstrapping.

3) *Robustness against reflective surfaces*: the HD^{4}AR is based on intensity-based image feature descriptors, such as SIFT, SURF, FREAK, or BRISK, which compare the intensity of pixels to discover orientation and response of feature points. As a consequence, the proposed approach may not work well with images that only show reflective surfaces such as metals, mirrors, or glass curtain walls of the building. These surfaces reflect all surrounding scenes and make the system difficult to find correspondences among the images. One possible method to address this is to require images to be taken farther from these elements so other non-reflective elements can also be presented in the scene.

Bibliography

[1] Golparvar-Fard, M., Peña-Mora, F., and Savarese, S., "Automated model-based progress monitoring using unordered daily construction photographs and IFC as-planned models," *ASCE Journal of Computing in Civil Engineering,* 2012. (in press)

[2] Allen, M., Regenbrecht, H., and Abbott, M., "Smart-phone augmented reality for public participation in urban planning," in *Proceedings of the 23rd Australian Computer-Human Interaction Conference*, pp. 11-20, 2011.

[3] Irizarry, J., Masoud G., Graceline W., and Bruce N. W., "InfoSPOT: A mobile Augmented Reality method for accessing building information through a situation awareness approach," *Automation in Construction*, vol. 33, pp. 11-23, 2013.

[4] Shin, D. H. and Dunston, P. S., "Identification of application areas for augmented reality in industrial construction based on technology suitability," *Automation in Construction*, vol. 17, no.7, pp. 882-894, 2008.

[5] Wang, X., "Improving human-machine interfaces for construction equipment operations with mixed and augmented reality," *Robotics and Automation in Construction*, pp. 211-224, 2008.

[6] Woodward, C. and Hakkarainen, M., "Mobile augmented reality system for construction site visualization," in *Proceedings of the 10th IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pp. 1-6, 2011.

[7] Behzadan, A. H. and Kamat, V. R., "Georeferenced registration of construction graphics in mobile outdoor augmented reality." *Journal of Computing in Civil Engineering*, vol. 21, no. 4, pp. 247-258, 2007.

[8] Izkara, J. L., Pérez, J., Basogain, X., and Borro, D., "Mobile augmented reality, an advanced tool for the construction sector," in *Proceedings of the 24th CIB W78 Conference on "Bringing ICT Knowledge to Work"*, 2007.

[9] Khoury, H. and Kamat, V.R., "High-precision identification of contextual information in location-aware engineering applications," *Advanced Engineering Informatics*, vol. 23, no. 4, pp. 483-496, 2009.

[10] Woodward, C., Hakkarainen, M., Korkalo, O., Kantonen, T., Aittala, M., Rainio, K., and Kähkönen, K., "Mixed reality for mobile construction site visualization and communication," in *Proceedings of the 10th International Conference on Construction Applications of Virtual Reality (CONVR)*, pp. 35-44. 2010.

[11] Gotow, J. B., Zienkiewicz, K., White, J., and Schmidt, D. C, "Addressing challenges with augmented reality applications on smartphones," in *Mobile Wireless Middleware, Operating Systems, and Applications*, pp. 129-143. Springer Berlin Heidelberg, 2010.

[12] Akula, M., Dong, S., Kamat, V. R., Ojeda, L., Borrell, A., and Borenstein, J., "Integration of infrastructure based positioning systems and inertial navigation for ubiquitous context-aware engineering applications," *Advanced Engineering Informatics*, vol. 25, no. 4, pp. 640-655, 2011.

[13] Ojeda, L. and Borenstein, J., "Personal dead-reckoning system for gps-denied environments," in *Proceeding of the 2007 IEEE International Workshop on Safety, Security and Rescue Robotics (SSRR)*, pp. 1-6, 2007.

[14] Bae, H., Golparvar-Fard, M., and White, J., "Enhanced HD^{4}AR (Hybrid 4-Dimensional Augmented Reality) for Ubiquitous Context-aware AEC/FM Applications," in *Proceedings of the 12th International Conference on Construction Applications of Virtual Reality (CONVR)*, pp. 253-262, 2012.

[15] Bae, H., Golparvar-Fard, M., and White, J., "High-precision and Infrastructure-independent Mobile Augmented Reality System for Context-Aware Construction and Facility Management Applications," in *Proceeding of the 2013 ASCE International Workshop on Computing in Civil Engineering (IWCCE)*, pp. 637-644, 2013.

[16] Bae, H., Golparvar-Fard, M., and White, J., "High-precision vision-based mobile augmented reality system for context-aware architectural, engineering, construction and facility management (AEC/FM) applications," *Visualization in Engineering*, vol. 1, no. 3, pp. 1-13, 2013.

[17] Bae, H., Golparvar-Fard, M., and White, J., "Rapid Image-based Localization using Clustered 3D Point Cloud Models with Geo-Location Data for AEC/FM Mobile Augmented Reality Applications," in *Proceedings of the International Conference on Computing in Civil and Building Engineering (ICCCBE)*, 2014.

[18] Bae, H., Golparvar-Fard, M., and White, J., "Image-based Localization and Content Authoring in Structure-from-Motion Point Cloud Models for Real-time Field Reporting Applications," *Computing in Civil Engineering*, 2014 (in press).

[19] Arth, C. and Schmalstieg, D., "Challenges of Large-Scale Augmented Reality on Smartphones," *the 10th IEEE International Symposium on Mixed and Augmented Reality Workshop (ISMAR)*, October 26, 2011, Basel, Switzerland.

[20] Chen, Y. and Kamara, J. M., "A framework for using mobile computing for information management on construction sites," *Automation in Construction*, vol. 20, no. 7, pp. 776-788, 2011.

[21] Feng, C. and Kamat, V. R., "Augmented reality markers as spatial indices for indoor mobile AEC/FM applications," in *Proceedings of the 12th International Conference on Construction Applications of Virtual Reality (CONVR)*, pp. 235-242, 2012.

[22] Lee, S. and Akin, Ö., "Augmented reality-based computational fieldwork support for equipment operations and maintenance," *Automation in Construction*, vol. 20, no. 4, pp. 338-352, 2011.

[23] Yabuki, N., Miyashita, K., and Fukuda, T., "An invisible height evaluation system for building height regulation to preserve good landscapes using augmented reality," *Automation in Construction*, vol. 20, no. 3, pp. 228-235, 2011.

[24] Hakkarainen, M., Woodward, C., and Billinghurst, M., "Augmented assembly using a mobile phone," in *Proceedings of the 7th IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR)*, pp. 167-168, 2008.

[25] Carozza, L., Tingdahl, D., Bosché, F., and Gool L., "Markerless vision-based augmented reality for urban planning," *Computer-Aided Civil and Infrastructure Engineering*, vol. 29, no. 1, pp. 2-17, 2012.

[26] Davison, A. J., Reid, I. D., Molton, N. D., and Stasse, O., "MonoSLAM: real-time single camera SLAM," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 29, no. 6, pp. 1052-1067, 2007.

[27] Dong, Z., Zhang, G., Jia, J., and Bao, H., "Keyframe-based real-time camera tracking," in *Proceeding of the 12th IEEE International Conference on Computer Vision (ICCV)*, pp. 1538-1545, 2009.

[28] Klein, G. and Murray, D., "Parallel tracking and mapping for small AR workspaces," in *Proceedings of the 6th IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR)*, pp. 225-234, 2007.

[29] Salas-Moreno, R. F., Newcombe, R. A., Strasdat, H., Kelly, P. H., and Davison, A. J., "SLAM++: Simultaneous localisation and mapping at the level of objects," in *Proceedings of the 2013 IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1352-1359, 2013.

[30] Wagner, D., Reitmayr, G., Mulloni, A., Drummond, T., and Schmalstieg, D., "Real-time detection and tracking for augmented reality on mobile phones," *Visualization and Computer Graphics, IEEE Transactions on*, vol. 16, no. 3, pp. 355-368, 2010.

[31] Ufkes, A. and Fiala, M., "A Markerless Augmented Reality System for Mobile Devices," in *Proceedings of the 2013 IEEE International Conference on Computer and Robot Vision (CRV)*, pp. 226-233, 2013.

[32] Triggs, B., McLauchlan, P. F., Hartley, R. I., and Fitzgibbon, A. W., "Bundle adjustment – a modern synthesis," in *Vision Algorithms: Theory and Practice*, pp. 298-372, Springer Berlin Hiedelberg, 2000.

[33] Gordon, I. and Lowe, D. G., "What and where: 3D object recognition with accurate pose," in *Toward category-level object recognition*, pp. 67-82, Springer Berlin Heidelberg, 2006.

[34] Irschara, A., Zach, C., Frahm, J. M., and Bischof, H., "From structure-from-motion point clouds to fast location recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2599-2606, 2009.

[35] Lim, H., Sinha, S. N., Cohen, M. F., Uyttendaele, M., "Real-time image-based 6-dof localization in large-scale environments," in *Proceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1043-1050, 2012.

[36] Sattler, T., Leibe, B., and Kobbelt, L., "Fast image-based localization using direct 2D-to-3D matching," in *Proceedings of the 13th IEEE International Conference on Computer Vision (ICCV)*, pp. 667-674, 2011.

[37] Agarwal, S., Snavely, N., Simon, I., Seitz, S. M., and Szeliski, R., "Building rome in a day," in *Proceedings of the 12th IEEE International Conference on Computer Vision (ICCV)*, pp. 72-79, 2009.

[38] Frahm, J., Fite-Georgel, P., Gallup, D., Johnson, T., Raguram, R., Wu, C., Dunn, E., Clipp B., Lazebnik, S., and Pollefeys, M., "Building Rome on a cloudless day," in *Computer Vision-ECCV 2010*, pp. 368-381, Springer Berlin Heidelberg, 2010.

[39] Snavely, N., Seitz, S. M., and Szeliski, R., "Modeling the World from Internet Photo Collections," *International Journal of Computer Vision*, vol. 80, no. 2, pp. 189-210, 2007.

[40] Lee, S. W., Jung J., Hong J., Lee S., Cho H., and Yang H. S., "Efficient 3D content authoring framework based on mobile AR," in *Proceedings of the 12th IEEE International conference on Visual Systems and Multimedia (VSMM)*, pp. 95-102, 2012.

[41] Tano S., Pei B., Ichino J., Hashiyama T., and Iwata M., "An architecture for ubiquitous and collaborative 3D position sensing for ubiquitous 3D drawing," in *Proceedings of the 15th IEEE International Conference on Computational Science and Engineering (CSE)*, pp. 469-476, 2012.

[42] Lowe, D.G., "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91-110, 2004.

[43] Bay, H., Ess, A., Tuytelaars, T., and Gool, L.V., "Speeded-Up Robust Features (SURF)," *Computer Visioin and Image Understanding*, vol. 110, no. 3, pp. 346-359, 2008.

[44] Leutenegger, S., Chli, M., and Siegwart, R.Y., "BRISK: binary robust invariant scalable keypoints," in *Proceedings of the 13th IEEE International Conference on Computer Vision (ICCV)*, pp. 2548-2555, 2011.

[45] Alahi, A., Ortiz, R., and Vandergheynst, P., "FREAK: fast retina keypoint," in *P roceedings of the 2012 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 510-517, 2012.

[46] Muja, M. and Lowe, D.G., "Fast matching of binary features," in *Proceedings of the 9th IEEE Conference on Computer and Robot Vision (CRV)*, pp. 404-410, 2012.

[47] Muja, M. and Lowe, D.G., "Fast approximate nearest neighbors with automatic algorithm configuration," in *Proceeding of the International Conference on Computer Vision Theory and Applications (VISAPP)*, pp. 331-340, 2009.

[48] Hartley, R. I. and Zisserman, A., *Multiple View Geometry in Computer Vision*. Cambridge University Press, 2004.

[49] Nistér, D., "An efficient solution to the five-point relative pose problem," *Pattern Analysis and Machine Intelligence*, *IEEE Transactions on*, vol. 26, no. 6, pp. 756-777, 2004.

[50] Hartley, R. I. and Sturm, P., "Triangulation," *Computer Vision and Image Understanding*, vol. 68, no. 2, pp. 146-157, 1997.

[51] Wu, C., Agarwal, S., Curless, B., and Seitz, S. M., "Multicore bundle adjustment," in *Proceedings of the 2011 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3057-3064, 2011.

[52] Furukawa, Y., Curless, B., Seitz, S. M., and Szeliski, R., "Towards internet-scale multi-view Stereo," in *Proceedings of the 2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1434-1441, 2010.

[53] Furukawa, Y. and Ponce, J., "Accurate, dense, and robust multi-view stereopsis," *Pattern Analysis and Machine Intelligence*, *IEEE Transactions on*, vol. 32, no. 8, pp. 1362-1376, 2010.

[54] Ward, J. H., “Hierarchical grouping to optimize an objective function,” *Journal of American Statistical Association*, vol. 58, no. 301, pp. 236-244, 1963.

[55] Norušis, M., *IBM SPSS Statistics 19 Statistical Procedures Companion*, Addison Wesley, 2011

[56] MacQueen, B., “Some methods for classification and analysis of multivariate observations,” in *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, pp. 281-297, 1967.