UNIVERSIDAD POLITÉCNICA DE MADRID

ESCUELA TÉCNICA SUPERIOR
DE INGENIEROS DE TELECOMUNICACIÓN

**A NEW EFFICIENT POSE ESTIMATION AND TRACKING
METHOD FOR PERSONAL DEVICES.
APPLICATION TO INTERACTION IN SMART SPACES**

TESIS DOCTORAL

JUAN LI

Ingeniera en Automatización y Control

2016

Departamento de Señales, Sistemas y Radiocomunicaciones

ESCUELA TÉCNICA SUPERIOR
DE INGENIEROS DE TELECOMUNICACIÓN
UNIVERSIDAD POLITÉCNICA DE MADRID

**A NEW EFFICIENT POSE ESTIMATION AND TRACKING
METHOD FOR PERSONAL DEVICES.
APPLICATION TO INTERACTION IN SMART SPACES**

TESIS DOCTORAL

Autora:

JUAN LI

Ingeniera en Automatización y Control

Director:

JOSÉ RAMÓN CASAR CORREDERA

Doctor Ingeniero de Telecomunicación

2016

Department: Señales, Sistemas y Radiocomunicaciones
Escuela Técnica Superior de Ingenieros de Telecomunicación
Universidad Politécnica de Madrid (UPM)

Ph.D. Thesis: A New Efficient Pose Estimation and Tracking Method for Personal Devices.
Application to Interaction in Smart Space

Author: Juan Li

Advisor: José Ramón Casar Corredera

Year: 2016

Committee named by the Rector of Universidad Politécnica de Madrid, on the of
..... 201 ..:

President:

Member:

Member:

Member:

Secretary:

After the defense of the Ph.D. Thesis on the of 201 .., at the
E.T.S.I. de Telecomunicación, the committee agrees to grant the following qualification:

.....

President

Members

Secretary

Acknowledgments

Firstly, I would like to express my special appreciation and thanks to my advisor Prof. José Ramón Casar Corredera for his continuous support of my Ph.D study. His immense knowledge, valuable experience, and foresight have directed the work in the thesis. His patience, suggestions, and funding have ensured the research went smoothly. I am grateful to him for providing me the opportunity to do my Ph.D with him. He has been a great advisor and mentor for my Ph.D study.

My sincere thanks also goes to Prof. Wilfried Philips from Ghent University and Prof. Hamid Aghajan from both Stanford University and Ghent University for their kindness and financial support, who provided me an opportunity to join their group TELIN in Belgium as a visiting scholar in 2015. The cooperation with this group was a nice experience, where I broaden my knowledge of other research topics. Also, the enthusiasm that Prof. Philips has for his research was contagious and motivational for me.

Besides, I would like to thank Dr. Paula Tarrío Alonso, who had co-advised me for the first two years. She gave me a lot of detailed suggestions on the research. She had been always patient with the problems that I encountered, regardless of big or small issues.

I must also thank another two professors from the group GPDS in Universidad Politécnica de Madrid: Prof. Juan Alberto Besada and Prof. Ana M. Bernardos. Their enormous knowledge on data filtering, data fusion, and smart space services has contributed a lot to the work in this thesis and some publications.

I appreciate a lot the comments and suggestions from the three experts that reviewed my thesis: Prof. Bart Goossens from Ghent University, Dr. Paula Tarrío Alonso and Prof. Ana M. Bernardos. Their insightful feedback are very helpful and valuable for improving the thesis.

Of course, I must express my gratitude to my lovely colleagues and ex-colleagues from the group GPDS in Universidad Politécnica de Madrid and the group TELIN in Ghent University, for all the fun we have had in the last five years, for the unforgettable days that we have passed, and for their kindness of being my Spanish teachers.

In addition, I would like to acknowledge the financial support from Chinese Scholarship

Council, Universidad Politécnica de Madrid, Ghent University and Consejo Social. Without these fundings, the research would not have been possible.

Finally, my appreciations will be given to my family for the support they provided me through my entire life and my friends for their encouragement.

Resumen

Esta tesis aborda la estimación y el seguimiento de posición y orientación de los dispositivos personales con seis grados de libertad (6-GdL) y las aplicaciones en espacios inteligentes. Este problema ha atraído la atención de industrias e investigadores de diversos campos, tales como los espacios inteligentes, la robótica, el seguimiento en interiores y la Realidad Aumentada. Además, se discute el problema relevante de la selección de la cámara en un sistema con múltiples cámaras, ya que es de fundamental importancia para la gestión de la cámara en una red de cámaras de gran tamaño.

A pesar de los grandes esfuerzos de investigación que se han llevado a cabo para hacer frente a estos problemas en los últimos años, sigue siendo un reto fundamental proporcionar un sistema de estimación de pose de bajo coste, preciso, rápido, fácil de implementar, robusto y que además sea adecuado para pequeñas y grandes áreas. Los sistemas existentes por lo general no pueden proporcionar una solución integral de interior teniendo en cuenta todos estos aspectos.

Para abordar estas cuestiones, esta tesis describe un sistema de múltiples sensores para la estimación de la posición exacta. El sistema se basa en tecnologías de bajo coste, en particular, en una combinación de uno o más sensores de visión externa, acelerómetros incorporados en el dispositivo y un marcador imprimible de color pegado en el dispositivo. Un conjunto de cámaras de infraestructura se despliegan para tener el objeto visible la mayor parte del tiempo de funcionamiento. El objeto tiene que incluir un acelerómetro de tres ejes incorporado y ser etiquetado con un marcador de referencia. El marcador está diseñado para que su detección sea fácil y robusta. Se puede adaptar mediante variaciones en forma y color a diferentes escenarios de servicio, como el seguimiento de dispositivos móviles, personas y robots. Con la ayuda de los acelerómetros, el sistema puede estimar la posición y la orientación con una o más cámaras basado en los enfoques propuestos de fusión de datos de múltiples sensores.

Dos algoritmos de seguimiento basados en el Filtro de Kalman son presentados con explicaciones detalladas de la aplicación, incluyendo la inicialización del filtro, el modelo del sistema dinámico, el ajuste de parámetros y el modelo de error de medición y de proceso. Un modelo de error del sistema completo se deriva analíticamente en base a la propagación de los errores. La secuencia de la innovación es explotada para detectar valores atípicos.

Además, se trata la falsa detección de valores atípicos debido al cambio de las fuentes de medición.

Se presenta asimismo un mecanismo de selección de la cámara en una red multi-cámara. En primer lugar, el enfoque selecciona cámaras disponibles que van a ver el objeto en el siguiente instante de tiempo basado en el estado predicho del sistema, la prueba point-in-view y la prueba de oclusión. Con respecto a la prueba de oclusión, se proponen varios métodos de modelado. Entonces, todas las cámaras disponibles se clasifican de acuerdo a una métrica de calidad: la distancia entre el objeto y la cámara.

Por otra parte, la tesis explora el potencial del sistema propuesto de seguimiento de posición y orientación en espacios inteligentes. Varios prototipos están diseñados en diversos campos, incluyendo las aplicaciones relacionadas con los apuntamientos, la realidad virtual para el aprendizaje de inmersión, los juegos en 3D y la realidad aumentada para la educación.

Los datos experimentales demuestran que el sistema propuesto de estimación de posición y orientación logra una alta precisión (del orden de centímetros para la estimación de la posición y de algunos grados para la estimación de la orientación), utilizando los sensores mencionados de bajo coste, trabajando en alrededor de 10 imágenes por segundo. De esta manera, se cumple el requisito de tiempo real de la mayoría de las aplicaciones. Los dos Filtros de Kalman propuestos son validados para ser coherentes, capaces de detectar valores atípicos y mantener la continuidad del seguimiento. Además, los resultados experimentales muestran que el enfoque propuesto de selección de la cámara proporciona una alta precisión y reduce en gran medida el coste computacional, especialmente en una red de cámaras de gran tamaño. En definitiva, se puede afirmar que el sistema propuesto es una solución precisa, rápida, robusta y fácil de implementar, de bajo coste y que tiene un gran potencial en el ámbito de los servicios en los espacios inteligentes.

Abstract

This thesis addresses a new efficient method for six-degree-of-freedom (6-DoF) pose estimation and tracking of personal devices and applications in smart spaces. This problem has attracted attention of industries and researchers from various fields such as smart spaces, robotics, indoor tracking, and Augmented Reality. Besides, the relevant problem of camera selection in a multi-camera system is also discussed, which is of fundamental importance to camera management in large camera networks.

Although major research efforts have been carried out to address these problems in recent years, it remains a critical challenge to provide a low-cost, accurate, fast, easy-to-deploy, and robust indoor pose estimation system, which is suitable for both small and large areas. Better said, existing systems usually fail to provide an indoor holistic solution taking into account all these aspects.

Addressing these issues, this thesis describes a multi-sensor system for accurate pose estimation that relies on low-cost technologies, in particular on a combination of one or more external vision sensor, embedded accelerometers in the device, and a printable colored fiducial to be stuck on the device. A set of infrastructure cameras are deployed to have the object to be tracked visible most of the operation time. The object has to include an embedded three-axis accelerometer and be tagged with a fiducial marker. The marker is designed to be easily and robustly detected. It may be adapted to different service scenarios (in shape and colors) such as mobile device tracking, person tracking, and robot localization. With the aid of accelerometers, the system can estimate the full pose with one or more cameras based on the proposed multi-sensor data fusion approaches.

Two tracking algorithms based on the Kalman filter are presented with detailed explanations of the implementation, including filter initialization, system dynamic model, parameter setting, measurement, and process error modeling. A complete system error model is analytically derived based on error propagation. The innovation sequence is exploited to detect outliers. Besides, the false detection of outliers due to camera hand-offs is dealt with.

A camera selection mechanism in multi-camera systems is presented. Firstly, the approach selects available cameras that will see the object at the next time instant based on the predicted state of the system, point-in-view test, and occlusion test. Regarding the occlusion

test, several modeling approaches are proposed. Then, all the available cameras are ranked according to a quality metric: the distance between the object and the camera.

Furthermore, the thesis explores the potential of the proposed pose tracking system in smart spaces. Several prototypes are designed in various fields, including pointing applications, Virtual Reality for immersive learning, 3D gaming, and Augmented Reality for education.

Experimental data demonstrates that the proposed pose estimation system achieves high accuracy (in the order of centimeters for the position estimation and few degrees for the orientation estimation) using the mentioned low-cost sensors, working at around 10 frames/sec, which fulfills the real-time requirement of most applications. The proposed two Kalman filters are validated to be consistent, able to detect outliers, and keep the tracking continuity. Also, experimental results show that the proposed camera selection approach provides high selection accuracy and largely reduces the computational cost, especially in a large camera network. All in all, we can claim that the proposed system is a low-cost, accuracy, fast, robust, and easy-to-deploy solution, being richly potential for services in smart spaces.

Contents

1	Introduction	1
1.1	Problem Description	4
1.2	Aims and Methodology	6
1.3	Summary of Contributions	10
1.4	Document Outline	11
2	Object Pose Estimation	13
2.1	Introduction	13
2.2	Related Work	15
2.2.1	Sensor-based Methods	16
2.2.2	Vision-based Methods	19
2.2.3	Hybrid Sensor-vision Methods	24
2.2.4	Summary of Sensing Technologies	25
2.3	Pose Estimation Strategy Overview	26
2.3.1	Notation, Coordinate Systems and Coordinate Transformations	26
2.3.2	Body Orientation Representation	28
2.3.3	Body Position Representation	29
2.3.4	System Description	30
2.4	Accelerometer as Inclination Sensor	31
2.4.1	Pitch and Roll Estimation	31
2.4.2	Accelerometer Calibration	32
2.5	Camera as Position Sensor	33
2.5.1	Fiducial Design	34
2.5.2	Fiducial Detection Algorithm	35
2.5.3	Projection of 3D Points into the Image Plane	37
2.6	Pose Estimation Using Fused Vision and Inertial Data	40
2.6.1	Data Synchronization	40
2.6.2	Monocular Vision Positioning System	41

2.6.3	Stereo Vision Positioning System	44
2.6.4	Multi-camera Solution by Triangulation	46
2.6.5	Constraint-based Occlusion Handling	47
2.6.6	Complete 6-DoF Pose Estimation	48
2.7	System Evaluation	49
2.7.1	Implementation and Prototype Setup	49
2.7.2	Testing Procedure	50
2.7.3	Accuracy Assessment	51
2.7.4	Computational Load Assessment	55
2.7.5	Discussions	55
2.8	Conclusions	57
3	Object Motion Tracking	59
3.1	Introduction	59
3.2	Related Work	62
3.2.1	Kalman Filter Framework	62
3.2.2	Review of Pose Tracking Techniques	64
3.3	Tracking Strategy Overview	67
3.3.1	Pose Tracker: System and Observation Models	67
3.3.2	Marker Tracker: System and Observation Models	68
3.3.3	Initialization	69
3.4	Measurement Error Modeling	69
3.4.1	Accelerometer Measurement Error Model	70
3.4.2	Reference Point Estimation Error Model	70
3.4.3	Determining Measurement Error Covariance Matrices	74
3.5	Process Error Modeling	78
3.5.1	Discretized Continuous-time Kinematic Models	78
3.5.2	Direct Discrete-time Kinematic Model	80
3.6	Outlier Detection	82
3.7	Experiments	84
3.7.1	Measurement Error Model Validation	84
3.7.2	Experimental Setup for Tracking Performance Analysis	85
3.7.3	Experimental Results	88

3.7.4	Discussions	96
3.8	Conclusions	97
4	Camera Selection in Multi-camera Systems	99
4.1	Introduction	99
4.2	Related Work	101
4.3	Determining Available Cameras	104
4.3.1	Point-in-view Test	104
4.3.2	Occlusion Test	106
4.4	Camera Selection Based on Quality Metrics	112
4.5	Camera Selection Performance Analysis	113
4.5.1	Experimental Setup	113
4.5.2	Model Accuracy Assessment	114
4.5.3	Estimation Accuracy Assessment	116
4.5.4	Execution Time Assessment	117
4.5.5	Discussions	118
4.6	Conclusions	118
5	Applications in Smart Spaces	121
5.1	Introduction	121
5.2	Service Integration	124
5.3	Pointing Applications	125
5.4	Virtual Reality for Immersive Learning	130
5.5	3D Gaming Based on Unity	132
5.6	Augmented Books	135
5.6.1	System Description	136
5.6.2	Performance Analysis	137
5.7	Conclusions	140
6	Conclusions and Future Research Directions	141
6.1	Conclusions	141
6.2	Future Research Directions	143
A	Pose Estimation for Augmented Books	147
	Bibliography	149

List of Figures

1.1	An example of remotely controllable devices in a smart home.	2
1.2	The complete system architecture.	7
2.1	Pose Estimation System hardware.	15
2.2	Inertial sensors.	17
2.3	Outside-in and inside-out tracking.	20
2.4	Active and passive markers and their applications in the state-of-the-art motion tracking systems.	22
2.5	Different types of markers.	23
2.6	Coordinate systems involved in the proposed pose estimation system.	27
2.7	Object orientation expressed in pitch-roll-yaw rotations.	28
2.8	Pose estimation system workflow.	30
2.9	The plot of accelerometer measurements in z axis when z-axis points up.	33
2.10	The colored and shape-based fiducial used in the proposed system.	34
2.11	The histograms in H, S, and V of the cyan color under distinct illuminations.	36
2.12	The histograms in H, S, and V of the cyan color under all three illuminations.	37
2.13	An example of image processing.	38
2.14	The pinhole camera model.	38
2.15	Camera calibration parameters.	39
2.16	The temporal description of each loop.	41
2.17	Monocular vision object positioning system.	42
2.18	Mono-view object positioning system geometry.	44
2.19	Stereo vision object positioning system.	45
2.20	The geometry of the occlusion handling system.	47
2.21	Sample images showing the results of the two pose estimation systems.	51
2.22	The average estimation errors for different distances between the mobile device and the external cameras.	53
2.23	Experimental setup	54

3.1	The ongoing discrete Kalman filter cycle.	63
3.2	Pose Tracker system architecture.	67
3.3	Marker Tracker system architecture.	68
3.4	A line segment expressed in the image.	71
3.5	Reference point error model simulation result.	73
3.6	Standard deviation (in pixels) for 1D quantification.	73
3.7	Uncertainty propagation model of the stereo vision system	76
3.8	Uncertainty propagation model of the monocular vision system.	76
3.9	χ_m^2 pdf with six degrees of freedom together with the 99% confidence region.	83
3.10	Standard deviation of pixel measurements from different distances to the cameras.	84
3.11	Comparison of position and Euler angles between experimental results and predicted results in the stereo vision system.	85
3.12	Comparison of position and Euler angles between experimental results and predicted results in the monocular vision system.	86
3.13	2D and 3D trajectories of the experiment.	86
3.14	The number of cameras that detect the marker on each frame.	87
3.15	The position and orientation estimation from four cameras.	87
3.16	Measurement residuals and the corresponding 3σ bounds from the Pose Tracker.	89
3.17	Measurement residuals and the corresponding 3σ bounds from the Marker Tracker.	90
3.18	Normalized innovation squared and the corresponding 99% confidence region	91
3.19	Autocorrelation of the innovation from the Pose tracker	91
3.20	Autocorrelation of the innovation from the Marker Tracker	91
3.21	Normalized innovation squared and the corresponding 99% confidence region after dealing with camera hand-offs.	92
3.22	Normalized innovation squared and the corresponding 99% confidence region after introducing some outliers.	92
3.23	The number of cameras that detect the marker on each frame when cam2 is deactivated.	93
3.24	The position and orientation estimation from three cameras.	94
3.25	A generated Region of Interest using prediction (blue rectangle).	96
4.1	The geometric relation of the user, the device, and the camera.	105
4.2	The pinhole model of a camera.	105

4.3	Rotate the normal vector $\mathbf{n}_b$ by ϕ	108
4.4	The spatial relation between a line l and a plane Π	109
4.5	The intersection of a ray and a plane.	110
4.6	A special case that the plane model does not work.	110
4.7	The rectangle and cuboid occlusion models.	111
4.8	The procedures in the camera selection module.	113
4.9	The complete processing loop.	113
4.10	An example of the views from the deployed cameras.	114
4.11	Camera selection results.	115
4.12	Captured images in the vicinity of the switch points.	116
5.1	The pose estimation system interface.	125
5.2	Arduino Uno and the WiFly shield.	126
5.3	Smart lamps and smart blind in the prototyped smart space.	127
5.4	The floor map of the Smart room prototype and the deployment of smart objects.	128
5.5	Graphic User Interface design.	129
5.6	System architecture in the smart space prototype.	129
5.7	A systematic description of the VR application.	130
5.8	An example of the virtual botanical garden.	131
5.9	Game scenario.	133
5.10	The setup of cameras and the gaming area.	134
5.11	The game areas.	134
5.12	Direction control of the player.	135
5.13	The world coordinate system in augmented books.	136
5.14	An example of a 3D virtual model popping up out of a page using the pro- posed approach.	137
5.15	Results from the augmented book experiment.	138
5.16	RMS errors for 400 frames with varying camera motion and varying distance.	139

List of Tables

2.1	Comparison of different types of markers.	23
2.2	Summary of sensing technologies.	25
2.3	Summary of variables in pose estimation.	27
2.4	Accelerometer calibration results.	33
2.5	Selected thresholds for magenta, cyan, and yellow color segments in the fiducial.	35
2.6	Errors in translation and orientation estimation from different distances and different orientations.	52
2.7	RMS of estimated position and rotation errors in the stereo vision system and the monocular vision system	54
2.8	Mean errors and variance of the two considered pose estimation systems. . .	55
2.9	Execution time of the system.	55
3.1	Errors of the original results and the filtered results from the Pose Tracker compared to the reference.	95
3.2	Errors of the original results and the filtered results from the Marker Tracker compared to the reference.	95
3.3	Average execution time for approaches with and without using ROIs.	96
4.1	Variables used in camera selection.	104
4.2	Average execution time for each method.	118
5.1	Smart objects and their functions.	127

List of Abbreviations

AR:	Augmented Reality
COTS:	Commercial Off-The-Shelf
DoF:	Degree of Freedom
EKF:	Extended Kalman Filter
HSV:	Hue Saturation Value
IMU:	Inertial Measurement Unit
KF:	Kalman Filter
LED:	Light Emitting Element
MEMS:	Micro-Electro-Mechanical Systems
NIS:	Normalized Innovation Squared
RF:	Radio Frequency
RFID:	Radio Frequency IDentification
ROI:	Region Of Interest
SIFT:	Scale-Invariant Feature Transform
SLAM:	Simultaneous Localization And Mapping
SURF:	Speeded-Up Robust Features
UKF:	Unscented Kalman Filter
UWB:	Ultra Wide Band
VR:	Virtual Reality
VSNet:	Visual Sensor Network

Chapter 1

Introduction

The term “ubiquitous computing”, introduced by Mark Weiser around 1988, describes the concept of merging computers seamlessly into the world to serve people in their daily lives at home and work (Weiser, 1991). It invoked an active research field - smart environments/spaces, which refer to the integration of various technologies and services to enrich the user experience and improve the quality of living. Context is a core concept in smart spaces, which was defined as “any information that can be used to characterize the situation of an entity. An entity is a person, place or object that is considered relevant to the interaction between a user and an application, including the user and applications themselves” in Abowd et al. (1999). With the ability of context awareness, the world can respond to users’ demands, provide task-relevant services/information and enhance the pleasantness of human-machine interaction. Nowadays, with the increasing availability of sensor networks, cheap computing power, and embedded systems as well as the advances in computer vision, human-machine interaction, and wireless networking, researchers have made much progress on smart space technologies.

Smart spaces have been involved in a wide range of application fields: smart home (e.g., for remote control, energy management, environmental monitoring, elderly care (Alam, Reaz, and Ali, 2012)), smart museums (e.g., to guide visitors through an unfamiliar space and provide additional information (Bruns et al., 2007)), smart cars (e.g., to detect the drowsiness of a driver (Sahayadhas, Sundaraj, and Murugappan, 2012)), etc. An example of a smart home is illustrated in Fig. 1.1, listing several remotely controllable devices.

In order to provide the aforementioned contextual services to its users, a smart space must be able to perceive the current state of the environment. Various affordable sensing technologies are available or under development, including activity recognition, multi-touch, microphone, etc. Depending on the information that is perceived, the sensing technologies are classified into five main categories: spatial, gesture and activity, environmental, voice, and hybrid sensing, as described in the following.

Spatial sensing: it involves sensing the location, orientation, or presence of the targets. Various techniques have been used for this task, including visual sensors, inertial sensors,

FIGURE 1.1: An example of remotely controllable devices in a smart home.

infrared, Bluetooth, Wi-Fi, etc. For example, in Bird and Arden (2011), self-contained inertial sensors and magnetometer are used for indoor navigation. With the advances in image acquisition and processing techniques, image sensors are playing an important role in many application fields, such as target tracking (Kim and Davis, 2006; Oh et al., 2011), robot navigation (Rampinelli et al., 2014), human-computer interaction (Rautaray and Agrawal, 2015), video monitoring and surveillance (e.g., in airports, parking lots, homes, etc. (Wang, 2013; Baghyasree, Janakiraman, and Parkavi, 2014)), health care (Lee and Chung, 2012), and smart spaces applications (Ronzhin, Prischepa, and Karpov, 2010). For example, they can monitor patients' daily behavior and activities to identify the cause of illnesses or provide suggestions for improving their lifestyle. Vision sensors are also widely used to detect markers or natural features in the images for object localization. For example, in Oh et al. (2011) and Martínez et al. (2011), a multi-camera system is applied to track an Unmanned Aerial Vehicle (UAV). Besides, radio-frequency positioning technologies also attract a lot of attention of researchers. For example, in Sinan, Zhi, Giannakis Georgios, et al. (2005), indoor accurate localization is achieved via ultra-wideband radios.

Gesture and activity sensing: gesture recognition enables users to interact with machines naturally by interpreting human gestures (e.g., hand gestures, facial expressions, and gazes). It is typically achieved by depth cameras. For example, Microsoft's Kinect V2 (*Kinect*) uses time-of-flight cameras to enable users to control and interact with the game console/computer through natural gestures. It is able to track the body silhouettes, fingers, gaze, and position. Alternatively, depth-sensing can be accomplished by stereo cameras. Examples include famous commercial products Sony PlayStation Camera (*Sony PlayStation*) and Leap Motion (*Leap Motion*). Leap Motion uses two infrared cameras to detect the hand and finger motions in a short range as input to substitute mice. Activity recognition such as walking, running, standing, and lying are important to analyze users' behaviors in a monitored area. It is typically achieved by wearable devices composed of motion sensors. For example, in Kwapisz, Weiss, and Moore (2011), users only have to carry the cell phones in their pockets.

Collected accelerometer data are used to identify the physical activity that a user is performing. In Wang et al. (2015), users are enabled to write certain commands in the air holding a mobile device to turn on/off the blinds, music, and TVs through analyzing accelerometers data.

Environmental sensing: various environmental sensors are used to monitor the parameters in the environment, such as light, temperature, humidity, smoke, gas, and barometric pressure. This is particular important in remote or dangerous environments. A list of examples of different environmental sensor network is provided by Hart and Martinez (2006).

Voice sensing: from a user's point of view, speech is a very straightforward manner of communication. Therefore, voice commands are attractive interaction approaches to users. For example, Kinect is integrated with speech recognition functionality. Nowadays, smartphones are mostly equipped with microphones and processing power, which may also perform the speech recognition task.

Hybrid sensing: a single input method may not be able to provide a holistic solution. Therefore, a variety of input methods can be combined. For example, Google Glass devices combine voice commands with a rim-mounted touch pad as the input. With equipped touchscreen, microphone, and cameras, mobile devices show vast potential in hybrid sensing.

Among the enabling technologies for smart spaces we can point out indoor spatial sensing, i.e., the object position and orientation estimation in 3D space. It is an important task in many traditional application fields. For example, in robotics, it is necessary to know accurately the position and orientation of the robot relative to the world coordinate system for robot guidance, object manipulation, etc. (Kyriakoulis and Gasteratos, 2010; Faessler et al., 2014; Huang et al., 2011). In recent years, mobile Augmented Reality (AR) has drawn a lot of attention of researchers and industries. AR relies on combining and superimposing virtual information over the real world, providing the user with extra computer-based information about resources, objects or points of interest. One of the crucial challenges is the registration of the virtual space with the physical space, referred to the need of real-time and accurate tracking of the hand-held device pose with respect to a certain coordinate system. The same need will exist for other types of device-centric interaction systems (Chaudhary et al., 2013), such as those based on pointing, in which the object to be controlled is selected by pointing at it with a cursor visualized through the screen in the user's device. Besides, it is of great importance to person and object tracking, such as for warehouse analytics, indoor drone tracking, or activity assessment.

The following section includes a problem description in Section 1.1. The aim of the thesis and the methodology are presented in Section 1.2. Section 1.3 summarizes the contributions of the thesis. Finally, Section 1.4 presents the structure of the thesis.

1.1 Problem Description

Typically, a smart space contains sensors, actuators, displays, and computational elements, which are connected and embedded in the everyday objects of our lives. In particular, with the development of mobile technologies, personal devices (e.g., smartphones, tablets, and PDAs) are playing an important role in smart spaces, which highlight in the following aspects:

- Mobility: mobile devices are often light-weighted and easily bound to users, i.e., carried or worn (e.g., glasses).
- Popularity: nowadays, mobile devices are widely owned by most people.
- Multi-function: mobile devices are typically equipped with a local processor, storage, cameras, sensors, a display screen, and wireless connection abilities for sensing, processing, communicating, and interacting.
- Personalization: devices are configured to a specific owner and secured for the access only by the owner.

Thanks to these features, mobile devices are widely adopted for smart space services to perceive contextual information and interact with the environment (Kwapisz, Weiss, and Moore, 2011; Wang et al., 2015; Li et al., 2016; Li et al., 2015c).

Let us imagine an environment populated with remotely controllable objects. In this space, indoor augmented reality interfaces may be used to complement the perception of the objects with virtual information and command controls. In order to provide a good user experience, these indoor AR interaction mechanisms will need the virtual information to be timely and perfectly located with respect to or superimposed over the reference resources in the real view. The user's visualization device may be a smartphone or a tablet, or a wearable object such as a pair of glasses or a helmet-like device. In any case, whatever the visualization device is, so as to place the virtual information correctly, it is necessary to accurately know its position and orientation within the 3D space.

In many scenarios that may be derived from the general one described above (e.g., museums, hotel rooms, smart offices, retails, etc.), the service providers may be willing to offer the users a low-cost complement to enable the successful operation of these interaction methods and other applications. From them, it can be derived that the pose estimation system has to fulfill the following requirements:

- Accuracy: high accurate pose estimation is necessary to align virtual information with the physical world. Otherwise, it may provide wrong augmented content. Centimeter level of accuracy is needed for a satisfactory indoor AR or pointing experience.

- Real time: the algorithm should be designed as low-cost in computation to provide timely information, with maximum allowed delays in the order of a quarter of a second.
- Simple infrastructure: the system should be easily and fast set up in a new scenario with as simple as possible infrastructure.
- Minimal modifications on the user side: we assume users are non-experts and use their own devices. In this case, the setup of the system in the user side should be easy to operate, e.g., in form of downloading an App and pasting an adhesive marker.
- Spatial scalability: the system should work in both small-scale and large-scale scenarios.
- Multi-user: the system should be able to localize multiple users and respond to each of them separately.
- Multi-service: the system should be a general solution for several application fields such as AR and robot tracking.
- Low-cost: the system should be based on low-cost technologies.
- Continuity: the continuity of the device's motion is maintained despite detection errors, occlusions, and the presence of other objects.

Three types of techniques are currently used to calculate the position and orientation of the device relative to the world: a) those based on sensors, b) those based on visual data and c) hybrid approaches. In the first group, specially, inertial sensors are popularly used, which are self-contained, fast, light-weighted, and compact. However, they suffer from severe drift problem due to the integration of measurement errors and require a periodic re-calibration from other sources to work reasonably. Various vision-based approaches have been proposed. A strength of visual sensors is the ability to provide rich information. In particular, the marker-based approaches have been widely used due to their ease-to-use and high accuracy. However, they meet challenges of scalability (e.g., markers have to be placed throughout the environment, they may interfere with the surrounding aesthetics if explicit ones, etc.), sensitivity to environmental changing conditions (e.g., in terms of illumination that may be altered due to numerous reasons) and sensitivity to perspective (the viewer can be trying to interact with the object at different distances and orientations thus making the detection difficult).

In order to estimate the state of the device over time despite occlusions, illumination changes, missed detections, outliers, it is necessary to filter the pose estimation results. Data filtering is task-orientated, depending on the measurement sensors, the process, the system model, the motions, etc. Filters based on the Bayes' probabilistic theory are typically used for this task (Bar-Shalom, Li, and Kirubarajan, 2004). They model the system using the state-space method and employ a recursive algorithm to incorporate new observations. These filters

are straightforward. However, the performance heavily depends on the design of the filter. Therefore, a concrete filter design and implementation are necessary for the pose estimation system.

In order for the system to work in a large-scale scenario, multiple cameras may be involved, which introduces some challenges in camera management, including sophisticated data fusion, increased cost on computation and communication. One active research direction is optimal cameras selection. That is to say, instead of processing all the images which may cause delay, only a subset of images will be processed. The selection mechanism varies with the camera network architectures. Some approaches may concern much about the network life time (Dagher, Marcellin, and Neifeld, 2006; Yu and Sharma, 2010), whereas others may focus on the data fusion quality (Park, Bhat, and Kak, 2006; Soro and Heinzelman, 2007). To our knowledge, there is much work about camera selection targeted at people tracking, but not at handheld device tracking, which differentiates from the former at: a) the dimension of a handheld device is much smaller than a person; b) it is easily occluded, especially by the user's body; c) the movement of a device is more flexible, having six degrees of freedom. To this end, a camera selection mechanism for taking handheld devices is on demand.

To sum up, despite the progresses that have been made to date, nowadays pose tracking systems still offer limited performance in terms of accuracy, computational cost, usability, robustness, scalability, and easiness of deployment. Better said, existing systems usually fail to provide an indoor holistic solution taking into consideration all these aspects: approaches delivering high accuracy are usually complex, expensive, difficult to scale, and may require bulky additional complements to tag the objects to track.

1.2 Aims and Methodology

In the context of the aforementioned challenges, the objective of this thesis is: a) to find a sound enabling approach to personal device pose estimation, which is low-cost, accurate, robust, fast, and scalable to both small and wide areas; b) to find a solution to filter the pose estimation data; c) to propose an effective camera selection mechanism in multi-camera systems; d) to study the potential application fields in smart spaces.

In the process of finding a sound enabling technology for six-degree-of-freedom pose estimation, we firstly developed a system based on the ultrasound location system and inertial sensors embedded on the mobile devices (Gómez et al., 2013). The ultrasound location system relies on a number of transmitters deployed in the ceiling and a receiver attached on the mobile device. When the receiver is under the coverage of at least three transmitters, the 3D position of the receiver can be triangulated. The orientation of the device, which transforms the device coordinate system into the world coordinate systems, is calculated by data acquired from the accelerometers and magnetometers embedded on the device. The experimental results showed that the ultrasound location system provided centimeter level

of accuracy. However, the orientation estimation based on the magnetometer was noisy indoor due to the influence from metallic objects in the surroundings.

We also proposed a laser-vision combined approach to indoor robot pose estimation in Li et al. (2017a). The approach combines one or two off-the-shelf laser pointers and a camera to provide a low-cost, fast and highly accurate solution. The laser pointers are mounted vertically in the robot and project points in the ceiling. The camera is deployed in a fixed position in the environment to observe the ceiling. The position and orientation of the robot are determined by detecting the projected laser points in the image. This approach achieved an accuracy of 14 mm in position estimation. However, it is only suitable for objects that move on the ground.

Taking into consideration the limitations of the previous work, the system that is finally proposed in this thesis is an accurate, fast and robust pose estimation and tracking system for personal devices (e.g., smartphone, tablet, glasses, etc.) by combining data from visual and inertial sensors. It is built on a simple apparatus, as is shown in the pose estimation module in Fig. 1.2: a) one or more vision sensors (commercial off-the-shelf cameras), which are fixed and calibrated beforehand, and deployed to have the tracked object under view, b) an accelerometer in the object to be tracked (i.e., accelerometers usually embedded in mobile devices) and c) a printable colored marker, to be stuck on the device.

FIGURE 1.2: The complete system architecture.

The video cameras are deployed in such a way that the device is seen by at least one of them at every position of the service space, thus they will be located in fixed locations usually at a reasonable height to avoid occlusions and oriented to have overlapped vision fields over the user's moving area. The color fiducial is designed to be quickly, easily, and efficiently located; it is ready to work in different illumination conditions, in order to be able to operate the system in real time and with a moving user. The thinness also allows them to be attached

to a small surface, for example, borders of mobile devices, hats or eyeglasses frames. In our approach, the marker is placed in the border of the device in such a way that it is clearly visible by the cameras. However, provided it is visible and well referenced to the device's geometry, it can be attached anywhere in the object.

Regarding the computation architecture, the pose estimation system will run under server-client configuration: the mobile client will send the accelerometers outputs to the server and these data will be fused with the information extracted from the vision sensors to estimate the device's pose. Once the pose is estimated, it will be delivered back to the client or to other subscribers through the established APIs, for the user services to exploit it. In the augmented reality application case, the user's device will contain an application to visualize the virtual information corresponding to the visible real objects over the image taken by the camera embedded in the user device. The position of the objects to be augmented on the screen will be calculated from the known 3D position of the objects and the 6-DoF pose estimated by the system. The infrastructure-centric approach avoids heavy computation on the consumer devices, saving device power, and gaining efficiency, although alternative architectures could be considered.

Let us now focus on the pose estimation strategy. In brief, the pose estimation algorithm of the proposed system consists of three steps. In the first step, the system captures the video streaming from the external cameras and implements a marker detection algorithm to locate the two reference points from each video streaming. At the same time, the device captures the accelerometer measurements and sends them to the server. The second step is to reconstruct the 3D positions of the reference points. Several algorithms are proposed depending on the number of cameras that detect the marker, including monocular vision positioning, occlusion handling algorithms, stereo vision positioning, and multi-camera solution, where the first two algorithms need the aid of accelerometers. The last step is to estimate position and orientation based on the reconstructed 3D positions of two reference points and the accelerometer measurements. The gravitational acceleration measured by the three-axis accelerometer contributes to estimate the device's inclination. Compared with pure vision-based approaches, the number of unknown parameters is reduced by the exploitation of accelerometer measurements.

The Kalman filter is an efficient and recursive filter to estimate the state of a linear system from a series of noisy measurements. The measurement and process uncertainties are expressed by the probability distributions. To keep tracking the mobile device over time despite detection errors, occlusions, and the presence of other objects, two tracking algorithms based on the Kalman filter are proposed and analyzed. The first Kalman filter is designed to track pose. As aforementioned in the pose estimation strategy, no matter how many cameras are involved, the algorithm will first reconstruct the 3D positions of the reference points. Therefore, the second filter is used to track the two reference points in 3D, pitch, and roll angles. The performance of filters heavily depends on the design and the parameter settings. To this end, a complete analytical error model for the pose estimation system is derived and validated by real tests. Based on this model, the measurement error covariance

matrix can be easily derived. The filter initialization, dynamic system model, and detection of outliers are explained in detail. The consistency of the filters are checked by analyzing the innovation sequence from a real test. The false detection of outliers caused by the change of measurement sources (camera switches) is dealt with.

The camera selection problem in a multi-camera network is divided to two sub-problems. The first problem is to look for the set of cameras that will see the target. This is achieved by the point-in-view test and the occlusion test. The point-in view test refers to checking if the projections of the predicted reference points fall into the image with known resolution. As is mentioned, the device is easily occluded by the human body. Therefore, the second test is to model the body and test whether the body will occlude the device. In the process, several occlusion modeling approaches are proposed, including the plane model, rectangle model, and others. The second problem is to select an optimal subset of cameras from those that will see the device. This problem can be considered in a game theoretic manner, as collaboration and competition coexist. Various quality metrics can be defined, whereas in this thesis, the distance between the marker and the camera is adopted according to the experimental observations. The proposed algorithm is evaluated in terms of model accuracy, estimation accuracy, and execution time. From the experimental results, it can be shown that the model accuracy is high and the computational cost is largely reduced while the estimation accuracy is maintained.

Finally, we study the potential applications of the proposed pose estimation system. The system is applied to a pointing application to remotely control smart devices in the space. It is also applied to Virtual Reality (VR) for immersive learning: a virtual 3D botanic garden is constructed and the user is allowed to walk around to have a 3D view of the plants from different perspectives and learn some additional information of the plants. Another example of application scenario is 3D gaming. In this application, the mobile device is programmed as a 3D controller which provides a 360-degree and immersive game experience. Finally, a derivation of the monocular pose estimation approach is proposed and applied to augmented books.

The complete system architecture is depicted in Fig. 1.2. There are four main modules: pose estimation, tracking, camera selection, and applications. The pose estimation module outputs the raw pose estimation results, which are noisy and may contain outliers. Then, the results are injected into the filter as observations. The filter provides an optimal estimate of the state of the dynamic system. On one side, the smoothed pose estimation results are sent to the camera selection module to select a subset of cameras to process for the next time instant in a multiple camera network. On the other side, they are delivered to the applications to provide pose-based services.

1.3 Summary of Contributions

The main contributions of this thesis can be summarized as follows:

Proposal of a linear colored marker. The marker has a linear thin stripe-like geometry, containing three colors. Compared to the traditional squared marker, the proposed linear marker may better adapt to final services, because the thinness allows it to be attached to a small surface such as borders of mobile devices and eyeglasses frames (Chapter 2).

Design and evaluation of a six-degree-of-freedom pose estimation system by fusing visual and inertial data. Several fusion algorithms are proposed, including a monocular vision system, a stereo vision system, a multi-vision system, and an occlusion handling algorithm. Especially, the pose estimation approach by combining one camera and the accelerometers largely reduces the density of the camera deployment. The performance of the proposed system is evaluated by real tests in terms of accuracy and execution time. The experimental results validate that the system is low-cost, robust, and accurate, being able to satisfactorily run in real time. Besides, the system can be customized to smart space services and scaled to both small and wide areas (Chapter 2).

Design and evaluation of two Kalman filters to track the motion of a mobile device. An analytical system error model is derived to analyze the stability of the system and estimate the measurement noise covariance matrix. The filters are able to detect outliers, predict the future states, and smooth results (Chapter 3).

Design and evaluation of a camera selection mechanism in a multi-camera system. To our knowledge, this is the first work addressing the camera selection problem for tracking mobile devices. The proposed approach is based on the predicted information from the Bayesian tracking scheme, geometrical constraints, body occlusion modeling and quality metrics. The approach is validated to be able to keep a seamless tracking of a mobile device with minimized cost on communication and computation (Chapter 4).

Implementation of several applications in smart spaces using personal devices. Four applications have been implemented. First, a smart space is installed with controllable and interactive objects. The personal device is used to remotely control smart objects in the environment by pointing at them. Another implemented application is Virtual Reality for immersive learning. Besides, an approach is proposed to apply the mobile device as a 360-degree immersive game controller. Furthermore, an algorithm derived from the monocular vision system, using the device internal camera and an external marker, is proposed. It is then applied to augmented books (Chapter 5).

1.4 Document Outline

The remaining chapters of this thesis are organized as follows:

Chapter 2 is devoted to object pose estimation approaches. In this chapter, different approaches to object pose estimation are reviewed. The strategy to fuse data from the target object's accelerometer with the input from one or more cameras is proposed. The performance of the proposed approach is further validated by experimental results in terms of estimation accuracy and execution time. It is compared to the ground truth and a reference marker-based system.

Chapter 3 describes the object tracking and prediction. Two Kalman filter-based algorithms have been proposed to track the motion of a mobile device. Both filters are able to keep the continuity of the motion, detect outliers, provide prediction information, and smooth results. A complete system error model is analytically derived to estimate the measurement error covariance matrix. The tracking performance of the proposed filters is evaluated by analyzing the innovation sequence from real tests.

Chapter 4 is dedicated to address the camera selection problem in a multi-camera system. A camera selection approach based on probabilistic techniques, geometric constraints, occlusion modeling, and quality metrics is processed. Its performance is validated by experimental results in terms of model accuracy, estimation accuracy, and execution time.

Chapter 5 is focused on the potential applications in smart spaces. In this chapter, we have applied the proposed handheld device pose estimation approach to several applications, including pointing applications, VR, 3D gaming, and augmented books. The prototypes have been implemented and analyzed.

Chapter 6 draws conclusions regarding the whole work. The proposals of future research lines are also included.

Chapter 2

Object Pose Estimation

2.1 Introduction

The term “pose” is usually employed to refer to the combined information on position and orientation of a moving target (i.e., an object or a human), being the acquired data referred to a reference coordinate system. Position is represented by the three-dimensional location of the object, while orientation may be expressed as a set of consecutive rotations. Determining the pose of a target in 3D space is an important task in many traditional application fields, such as robotics (Kyriakoulis and Gasteratos, 2010; Faessler et al., 2014; Huang et al., 2011) (e.g., for robot guidance, object manipulation, etc.), indoor tracking (Li et al., 2015a; Li et al., 2017b), activity estimation, and interaction (Chaudhary et al., 2013).

In particular, in recent years, an attractive application area requiring accurate pose estimation is Augmented Reality (AR) (Höllerer and Feiner, 2004; You, Neumann, and Azuma, 1999; Zhou, Duh, and Billingham, 2008; Föckler et al., 2005; Schall et al., 2009; Gómez et al., 2013; Wagner et al., 2010; Klein and Drummond, 2004; Bruns et al., 2007; Chen, Chang, and Huang, 2014). AR has been widely explored in training, entertainment, education, and tourism (Höllerer and Feiner, 2004) to facilitate a novel way for the users to interact with their surroundings. Ideally, an AR system should be able to overlay the virtual information upon the real world with no error and no latency, thus it needs a perfectly estimated pose of the visualization object relative to the real world. When indoors, current technologies are not able to achieve these performance goals (very accurate, low-cost estimation of position continues to be a challenge and inertial systems drifts are still a limitation), thus a variety of pose estimation technologies are being developed by researchers to make a compromise among accuracy, cost, robustness, computational complexity, and on-board power consumption.

A number of pose estimation solutions have been already built on visual sensors. A strength of visual sensing technology is that it can provide high accuracy. However, at the same time, these solutions are computationally complex and require an ad-hoc infrastructure. An alternative is to use inertial sensors, which are self-contained and provide high measurement rates, therefore being able to track fast movements. Moreover, with the evolution of MEMS

(Micro-Electro-Mechanical Systems), inertial sensors have largely reduced their price and size, and they are increasingly embedded in mobile devices and smart objects. On the down side, inertial systems suffer from drift through acceleration measurement integration. Taking these facts into account, a sound approach can be to design a multisensor system: the advantage of fusing data from different sensors is that the limitations of a sensor type can be compensated by another one, thus better performance can be achieved with respect to one single sensor solution. Several approaches for integrating visual and inertial sensing technologies have been proposed in the literature (Kyriakoulis and Gasteratos, 2010; You, Neumann, and Azuma, 1999; Kelly and Sukhatme, 2011; Ligorio and Sabatini, 2013; Satoh, Uchiyama, and Yamamoto, 2004), where the drift problem is alleviated but not eliminated. This thesis overcomes this problem by adopting only gravitational acceleration measured by inertial systems for pose estimation. As no acceleration integration process is performed, the solution proposed in this thesis is zero-drift.

Following this approach, this chapter describes a multisensor solution for accurate pose estimation by combining inertial and vision sensors. Our goal has been to develop a low-cost, robust, and accurate pose estimation system able to satisfactorily run in real time, being customizable to smart space services and scalable for both small and wide areas. The multisensor system that has been designed provides high accuracy while relying on low-cost technologies, in particular on a combination of webcams and printable colored fiducials. This makes the system a cheaper alternative to commercial fixed camera 6-DoF trackers, such as Opti-Track (*Optitrack*) and Vicon (*Vicon Motion Capture System*). Our novel hybrid 3D pose tracker is targeted at general purpose rigid objects (deformable objects are beyond the scope of this thesis).

The enabling apparatus is simple, as depicted in Fig. 2.1, a) one or more infrastructure vision sensors (commercial off-the-shelf cameras), which are fixed and calibrated beforehand, b) a three-axis accelerometer in the object to be tracked (e.g., embedded accelerometers in mobile devices), c) a printable colored marker to be stuck on the object and d) a server. The pose calculation process is implemented on the server side, leaving computing power of the client side for applications.

Within a bounded space, the system can work with a single active camera (monocular approach), with two cameras (stereo vision approach) or with multiple cameras (multi-camera approach). To equip a room-like space with our pose estimation technology, more than two cameras may be needed to cover the whole space; although the multi-camera deployment design should aim to provide stereo vision along the room, the system will be able to work even if a single camera only covers certain areas. The issues related to multi-camera management, such as object tracking and camera selection, will be studied in the following chapters. In this chapter, we concentrate on describing the strategy to fuse data from the target object's accelerometer with the input from one or two or multiple cameras. From our experimental results, it will be shown that the proposed pose estimation system has great potential in practical applications, as it achieves high accuracy (in the order of centimeters

FIGURE 2.1: Pose Estimation System hardware.

for the position estimation and few degrees for the orientation estimation) using the mentioned low-cost sensors.

The rest of the chapter is organized as follows. Section 2.2 includes a review of previous work on object pose estimation systems. The problem description is provided in Section 2.3. Section 2.4 explains the contributions of accelerometers as inclinations sensor, while Section 2.5 describes the contribution of cameras as position sensor. Section 2.6 is dedicated to describe the pose estimation strategy, which fuses data from inertial and vision sensors. Experiments are assessed in Section 2.7. Section 2.8 concludes this chapter with further lines of work.

2.2 Related Work

Object pose estimation has been studied over the past several decades and a wide range of technologies have been explored (Kyriakoulis and Gasteratos, 2010; You, Neumann, and Azuma, 1999; Kelly and Sukhatme, 2011; Ligorio and Sabatini, 2013; Satoh, Uchiyama, and Yamamoto, 2004; Faessler et al., 2014; Huang et al., 2011; Zhang, Fronz, and Navab, 2002; Zhou, Duh, and Billingham, 2008; Föckler et al., 2005; Höllerer and Feiner, 2004; Kovavisaruch et al., 2012; Schall et al., 2009; Fiala, 2005; *ARToolKit* 2010; Gómez et al., 2013; Jamal, 2012; Wagner et al., 2010; Olson, 2011; Klein and Drummond, 2004; Henry et al., 2012; Newcombe et al., 2011; Bruns et al., 2007; Chen, Chang, and Huang, 2014; Feldman et al., 2005), such as GPS, inertial sensors, magnetic sensing, and optics. Until now, determining the position and orientation of an object is still a complex problem with no single best solution. Each of these technologies has its advantages and limitations. Depending on the sensing

technology, the available approaches may be classified into three main categories: sensor-based, vision-based, and hybrid approaches. The existing literature on these categories is described in the following.

2.2.1 Sensor-based Methods

According to work modalities, sensor-based methods can be divided into inertial, magnetic, electromagnetic, ultrasonic, and radio based categories.

A. Inertial sensors

Inertial sensors including accelerometers and gyroscopes, can be used as a dead reckoning system to provide continuous estimations of the position, velocity, and orientation of a mobile object. They have been widely used for robots (Kyriakoulis and Gasteratos, 2010), aircrafts, and vehicles navigation (Jamal, 2012). Inertial Measurement Units (IMU) are maturely developed units for motion tracking which typically contain three orthogonal accelerometers and three orthogonal gyroscopes, as shown in Fig. 2.2a. The principle for determining position and orientation using these sensors is based on Newton's laws. Gyroscopes measure the angular velocity and by integrating once, rotation angles can be calculated. This orientation can then be used to transform the acceleration measurements from the *body reference frame* (in which all the measurements are taken) into a *global reference frame*. Afterwards, accelerations due to gravity can be subtracted. The remaining acceleration, also called *linear acceleration* is integrated once to get the velocity and then integrated again to get the position estimation. This procedure is illustrated in Fig. 2.2b. Over the last few years, the evolution of MEMS technology has enabled the availability of low-cost and small size inertial sensors to be integrated into mobile devices (e.g., PDA, mobile phones, and tablet PCs), which facilitates the usage of inertial sensors.

Inertial sensors are self-contained, that is, they do not rely on other external resources. This advantage makes them particularly attractive for localization in unprepared environments without range limitation. They run at a high rate, therefore they are able to track fast and abrupt movements. Furthermore, they are not influenced by illumination and no line-of-sight is necessary. On the downside, as the new positions are calculated from the integration of previous measurements, measurement errors are cumulative and lead to an unbounded increase in the error of the estimated position, which is known as the severe drift problem. Thus, a periodic re-calibration is required. Several methods have been proposed to minimize the drift problem. For example, in Rolland, Davis, and Baillot (2001), relative measurements were used instead of absolute measurements to reduce the drift error. It is worth mentioning that in inertial-based methods, the initial state is needed to calculate the absolute pose.

FIGURE 2.2: Inertial sensors. (A) An Inertial Measurement Unit containing three accelerometers and three gyroscopes. (B) Functional flow diagrams for an inertial navigation system.

B. Magnetic sensors

Magnetic field sensing approaches are based on the principle of magnetic induction: when a coiled wire is moved through a magnetic field, an electrical current will flow in the coil. The strength of this current is a function of the distance and the orientation of the coil relative to the source of the magnetic field. A magnetic receiver in the natural magnetic field of the Earth is an one-DoF tracker, indicating the direction relative to the “magnetic north”. To achieve six-DoF pose estimation, an artificial magnetic field is required, which is typically generated by a transmitter containing three orthogonally orientated coils. Then, the position and orientation of a receiver in this field are deduced based on the induction. Magnetic trackers are lightweight, support multiple sensors and do not suffer from occlusion. Unfortunately, as is well-known, one problem of magnetic sensing is the sensitivity to magnetic and electrical interference caused by metallic objects within the operating volume. Additionally, it is limited in range due to the decay of the strength with the distance between the emitting source and the sensors (Franz et al., 2014). Under laboratory conditions, the

position can be estimated with several millimeter-level accuracy for a distance of a couple of centimeters (Wilson et al., 2007). Sixsense Entertainment, a leading company in magnetic motion tracking for video games, has increased the optimized performance range to 8-foot radius from the base with its latest technology (Khalfin and Rubin, 2013).

Magnetometers are used to get the heading angle by sensing the earth magnetic field (Schall et al., 2009; Bird and Arden, 2011). In order to get a full pose estimation, they need to be combined with other technologies. The algorithm in Bird and Arden (2011) integrated inertial sensors with magnetometers and kept the tracking results within about 2 m of the true track throughout the entire in-building run. However, the measurements provided by magnetometers can be corrupted due to the presence of metallic objects in the surroundings, which is quite usual in indoor scenarios (Martin et al., 2014).

C. Radio frequency technologies

Radio-frequency (RF) positioning systems aim at locating indoor moving objects (smartphones, robots, etc.) through diverse technologies and techniques (refer to e.g., Liu et al. (2007) and Sun et al. (2005) for a survey) based on times of arrival, received signal strength indicator, or phase difference of signals. WiFi, Bluetooth, or ZigBee-based solutions usually rely on fingerprinting techniques (e.g., Xiao, Ni, and Toh (2011)) or channel modeling (e.g., Tarrío, Bernardos, and Casar (2011)) to achieve a limited accuracy (3-4 meters in average); these positioning systems require, in general, an exhaustive calibration (fingerprinting requires building a RF map and channel modeling requires empirical estimation of diverse parameters), coming to be costly to deploy and maintain. Ultra-wideband (UWB) is able to provide centimeter-level accuracy (Sinan, Zhi, Giannakis Georgios, et al., 2005) using time of arrival methods. However, UWB signal is under the influence of the multi-path effect and non-line-of-sight conditions (Fan et al., 2014) and require specific infrastructure and in-device hardware, thus being complicated to universalize. Radio Frequency Identification (RFID)-based localization is a simple technology which gives a rough estimation of users' locations through the communication between RFID tags and RFID readers (e.g., Kodaka et al. (2008)), but dense and specific deployments are needed to cover the operation area (e.g., RFID mats or floors).

Regarding orientation estimation, RF positioning systems do not generally support or use it. Although the signal variability is noticeable in different orientations (Bahl, Padmanabhan, and Balachandran, 2000), introducing orientation-aware fingerprint or channel models multiplies the calibration efforts and does not provide sound results (Deyle, Kemp, and Reynolds, 2008). Hybrid strategies combining RF and inertial or magnetometers make possible both to improve location accuracy and retrieve orientation information. For example, WiFi can be fused with inertial information, achieving an average error in the order of 1.5 meters (e.g., Xiao, Ni, and Toh (2011) and Martin et al. (2014)). With a similar approach, UWB has also been fused with inertial navigation systems to achieve 6-DoF pose estimation (Fan et al., 2014; De Angelis et al., 2010). In Yut, Fei, and Geng (2013), a Zigbee network is combined with inertial sensors to improve the accuracy of UAV indoor localization. In

general, hybrid solutions do not provide enough accuracy for demanding applications (e.g., AR) or require specific hardware (e.g., UWB).

D. Ultrasonic-based

Ultrasonic tracking can estimate both pose and velocity information. They are typically composed of one base station, several transmitters and a receiver. The time-of-flight of the signal in its propagation from a transmitter to a receiver is used to calculate the distance between them with the knowledge of the propagation speed of sound. In order to calculate the 3-dimensional coordinates of the mobile mote, at least three beacons should be heard. A typical ultrasonic localization system is Cricket, which was used in the work in (Gómez et al., 2013) to track the position of a mobile device by attaching the mobile note to the device. The main advantages of using this kind of localization system are that the accuracy is high enough for indoor AR applications (in the order of few centimeters), that it is easy to use and manage, and that the costs due to computation and communication are much lower than for systems using computer vision. On the downside, the system requires line-of-sight paths between the beacons and the listener, and the deployments need to be dense, as the range of the ultrasound is very low (around 10 meters). Moreover, ultrasound sensors are sensitive to temperature, as the propagation speed of sound changes. According to this, this kind of localization system is appropriate for indoor environments without many obstacles. It could be used, for example, for guided tours in museums or interaction in smart spaces.

2.2.2 Vision-based Methods

Visual sensing technologies try to interpret the environment through observations from cameras. Most of the available proposals estimate the spatial relationship between the camera and the object by finding the correspondence between 2D image points and 3D scene points. According to the tracked features, most of the methods can be grouped into marker-based, ready to decode a known external visual reference, and markerless methods without need of any previously known symbol. According to the system configuration, methods can be classified as outside-in methods (or infrastructure-based methods), in which fixed cameras are installed to track the target object, and inside-out methods (or target-based methods), in which the scene is observed by an on-body camera.

A. Outside-in and inside-out tracking

Within the domain of optical tracking, outside-in systems are usually composed of wall- or tripod- mounted cameras at fixed locations directed towards the volume where the target would appear. Most commercial optical trackers use this configuration, such as ARTrack (*ARTrack*), OptiTrack, and Vicon. In these systems, the objects to be tracked are marked with active or passive landmarks. The system architecture is illustrated in Fig. 2.3a. The number and/or the shape of the landmarks needed on each object depend on the number of degrees of freedom. LEDs are mounted on the objects to be tracked in (Klein and Drummond, 2004;

Faessler et al., 2014), which are observed by an external camera. In (Li et al., 2017a), one or more laser pointers are mounted on a robot. The projection in the ceiling is detected by an external fixed camera to retrieve the position and orientation of the robot.

FIGURE 2.3: Outside-in and inside-out tracking. (A) Outside-in optical tracking. (B) Inside-out optical tracking.

Inside-out tracking systems determine the position and orientation of the device through its own mounted camera, as shown in Fig. 2.3b. By observing the scene which is marked with natural landmarks or artificial landmarks and finding the correspondences between 3D points and 2D points, the position and orientation of the mobile camera can be calculated. They are popularly developed for mobile device applications (e.g., Augmented Reality (Wagner et al., 2010)). In 1991, the University of North Carolina-Chapel Hill reported an optical ceiling tracking technology based on inside-out tracking (Azuma and Ward, 1991). It was accomplished by placing light emitting elements (LEDs) on the ceiling and placing an optical sensor cluster on the user's head. This system was designed to provide high accuracy over a large area. However, it is not practical in use because it requires a very dense array of bright infrared LEDs deployed in the ceiling. Another wide usage of inside-out tracking system is handheld Augmented Reality. The pose of a handheld device is determined by tracking artificial markers or natural features from the internal camera.

Both tracking configurations have advantages and disadvantages. The choice of the configuration is application-dependent. Inside-out tracking systems typically have simple infrastructures, as they do not rely on external cameras. However, in case of a mobile device, they have high consumption on on-board computation and battery. Outside-in tracking systems exhibit better scalability to multiple targets and in a wide area (Pintaric and Kaufmann, 2007) than inside-out self-trackers. Also, they provide a bird-eye view of the whole scene. The position of a landmark (such as an LED or a retro-reflective ball) can be obtained by triangulation with high accuracy. However, in an outside-in tracking system, only position estimates can be made directly, and so orientation is calculated from position estimates of multiple landmarks on the object. This leads to a fact that the quality of the orientation estimation is dependent on the quality of the position estimates and the baseline between the landmarks. Therefore, the system designer should keep the baseline of the landmarks as large as possible to reduce the orientation sensitivity to the position estimation error.

B. Marker-based methods

Marker-based methods recover the transformation between the fiducial (artificial) marker and the vision sensor by extracting the feature points previously defined in the marker. The objects to be tracked are normally marked with active, passive, or printable fiducial landmarks. Active markers are mostly infrared LEDs. They are easily detected and robust to illumination changes. Also, they provide visibility up to long distances within a wide angular range. They are used in several systems, including the state-of-the-art commercial products ARTrack and Vicon. An example of a finger tracking approach by using LEDs from ARTrack is shown in Fig. 2.4a. The system proposed in Klein and Drummond (2004) is designed for a tablet-based AR service. It relies on six LEDs mounted on the back of the tablet, which are then combined through outside-in and inside-out tracking methods to calculate the pose of the target. Obviously, the fact of having six LEDs attached to the device makes the system complex in terms of real service-oriented feasibility. Moreover, the infrastructure cameras have to see the whole back of the tablet, which largely constrains the movement of the tablet, making it unnatural for the user. Another LED-based system is described in Faessler et al. (2014), in which four LEDs are placed on a robot to enable its tracking. However, LEDs may be bulky and need to be powered either by wires or by batteries.

Passive markers are retro-reflective elements, i.e. they reflect the incoming infrared radiation into the direction of the incoming light, as shown in Fig. 2.4b. More precisely the infrared radiation is back-reflected into a narrow range of angles around the source of the incoming light. These markers are mostly spheres covered with retro-reflecting foils, but can also be stickers made from retro-reflecting material. They are used in two of the most famous motion tracking systems, as shown in Fig. 2.4c (Opti-Track (*Optitrack*)) and Fig. 2.4d (Vicon (*Vicon Motion Capture System*)). They use fixed high-speed infrared cameras following the outside-in approach and provide highly accurate results. However, they are expensive and not suitable for a low-cost and simple service. Furthermore, all cameras must be equipped with strobe light.

Printable fiducial markers are also widely used due to the low cost and ease to use. Several available libraries use planar fiducial for tracking, such as ARToolKit (*ARToolKit*), ARTag (Fiala, 2005), Studierstube (Schmalstieg and Wagner, 2009), AprilTag (Olson, 2011), and OpenCV (*OpenCV*). ARToolKit was developed in 1999 by Hirokazu Kato and has been widely used since then. Based on ARToolKit, ARTag was later developed to provide improved performance. The extended version of ARToolKit is ARToolKitPlus, which added more features over the ARToolKit. However, it is no longer developed and has a successor: Studierstube Tracker. It supports mobile phones as well as PCs and has low memory requirements. However, it is not open source. AprilTag has been recently developed for PCs and further improves accuracy and robustness. OpenCV is an open source cross-platform toolkit for image processing that supports PCs as well as mobile platforms. This library is still in development and has a large community of users. In Prochazka and Koubek (2011), a chessboard pattern is tracked by OpenCV to implement mobile AR. Apart from square

FIGURE 2.4: Active and passive markers and their applications in the state-of-the-art motion tracking systems. (A) Active markers used in ARTrack. (B) Passive markers. (C) OptiTrack. (D) Vicon.

markers, circle markers (Naimark and Foxlin, 2002) and 3D markers (Mohring, Lessig, and Bimber, 2004) were also proposed. Some examples of different types of markers are listed in Fig. 2.5. A comparative study of several visual markers can be found in Zhang, Fronz, and Navab (2002). Regardless of the shapes of the markers, markers used in these libraries are typically black-white and have high contrast, so they are easily recognizable. On the downside, contrast-based detection is sensitive to lighting. Color can also be encoded to increase the amount of information. In Grillo et al. (2010), barcodes with embedded color information are presented. Generally speaking, marker-based methods can provide high accuracy. However, the marker size and the distance as well as the viewing angle to the marker will affect the accuracy. These aforementioned markers need a big and flat surface to be placed. Therefore, they are unsuitable to be attached to a small object to be tracked, such as a mobile device. Instead, by using an on-body camera to track markers placed in known locations it is feasible to estimate the object's pose. These approaches are widely used in simple scenarios because of their easy setup. However, their use can be complicated in a wide working area. For example, large quantities of markers need to be deployed and measured carefully. Markers can be intrusive to the environment (causing visual discomfort). In case of a mobile device, the on-board imaging processing is battery-draining.

Finally, a comparison of the three types of markers is summarized in Table 2.1.

FIGURE 2.5: Different types of markers. (A) ARToolKit (4 examples). (B) ARTag (4 examples). (C) AprilTag (4 examples). (D) Circle markers proposed in Naimark and Foxlin (2002).

TABLE 2.1: Comparison of different types of markers.

Types	Advantages	Disadvantages
Active markers	Accurate, robust, long distance of visibility	Electric power either by wires or by batteries is needed
Passive markers	Accurate, simple	All cameras must be equipped with strobe light
Printable fiducial markers	Easy to deploy, cheap, many patterns	They are easily affected by the illumination changes

C. Markerless methods

In order to get rid of artificial markers, researchers are making efforts to detect natural landmarks from image sequences, which are also referred as markerless methods. Many robust local descriptors including SIFT (Scale-Invariant Feature Transform) (Lowe, 2004) and SURF (Speeded-Up Robust Features) (Bay, Tuytelaars, and Van Gool, 2006) are stable under different viewpoints and lighting conditions. They can be used to detect features existing in the scene to build a markerless method. However, their computational requirements are stronger than those relying on artificial markers. Although there are some proposals to

adapt them to mobile platforms (Wagner et al., 2010), the use of markerless methods is still a challenge for mobile devices with limited computational capabilities. Simultaneous Localization and Mapping (SLAM) systems have been recently developed to run on mobile devices in large-scale areas. For example, a key-frame based SLAM system that runs locally on a camera-equipped mobile device is presented in Ventura et al. (2014). A feature-less monocular SLAM algorithm is presented in Engel, Schöps, and Cremers (2014), which allows to build large-scale and consistent maps of the environment. A survey on SLAM can be found in Fuentes-Pacheco, Ruiz-Ascencio, and Rendón-Mancha (2015). Generally speaking, natural feature tracking is still a complex problem and is usually expensive in terms of computational cost, being challenging for hand-held devices.

With the advances in imaging technologies, new types of vision sensors have been developed. RGB-Depth cameras with moderate prices, such as Microsoft Kinect device (*Kinect*), have attracted attention of researchers. These devices capture RGB images along with per-pixel depth information. In Huang et al. (2011), a micro air vehicle is mounted with a Microsoft Kinect camera to track its pose. Similarly, authors in Newcombe et al. (2011) have designed a system composed by an infrared dot-pattern projector and an infrared camera; the system recovers the transformation by finding correspondences between the reference pattern and the detected dot grids. Time-of-flight cameras are relatively new sensors that deliver 3-dimensional imaging at a high frame rate, simultaneously providing intensity data and range information for every pixel (Foix, Alenya, and Torras, 2011).

2.2.3 Hybrid Sensor-vision Methods

Each of the previous approaches has its strengths and limitations. An alternative solution that tries to exploit their advantages, while softening their hindrances, is to combine different sensor types, as it is done in You, Neumann, and Azuma (1999), Kelly and Sukhatme (2011), Ligorio and Sabatini (2013), and Satoh, Uchiyama, and Yamamoto (2004). In our previous work (Gómez et al., 2013), data from an ultrasound location system were fused with the magnetic sensor of a tablet device to provide pose information for AR services. The ultrasound location system provided centimeter level of accuracy. However, the orientation estimation based on the magnetometer was noisy indoor due to the influence from metallic objects in the surroundings.

Typically, visual and inertial measurements are combined using a filtering framework. Kalman filter and its derivatives (Unscented Kalman filter, Extended Kalman filter) are favorably selected to perform sensor fusion by integrating measurements from the vision-based system and from the sensor-based system (Kelly and Sukhatme, 2011; Ligorio and Sabatini, 2013).

Most of the hybrid systems choose gyroscopes for calculating the orientation through integration. For example, within the tracking methods described in You, Neumann, and Azuma (1999) and Satoh, Uchiyama, and Yamamoto (2004), a gyroscope was adopted to measure the orientation, and in return, the vision-based system was focused on correcting the drift of

TABLE 2.2: Summary of sensing technologies.

Technol- ogy	Description	Strengths	Weaknesses
Inertial	Use accelerometers and gyroscopes. Gyroscopes measure the angular rates, which are integrated to calculate the rotation angles. Accelerometers measure the body accelerations, which are integrated twice to estimate position.	Fast Self-contained No line-of-sight problem No range limitation Small size	Severe drift Initial status is needed
Magnetic	A transmitter composed of sets of coils generates an artificial magnetic field. The receiver senses the distance and the orientation relative to the source of the magnetic field.	Accurate Lightweight No line-of-sight problem Not affected by illuminations Good noise resistance	Ferromagnetic and/or metal conductive surfaces cause field distortion Accuracy diminishes with distance
Radio frequency based	Estimate the distance between the receiver and the transmitter based on the received signal strength, angle of arrival or time difference of arrival. The 3D position of the target is estimated via triangulation.	Inexpensive because the existing infrastructure can be used, such as Wi-Fi hotspot Can achieve large-scale localization	Multi-path problem Low accuracy Radio noise interference
Ultra-sound	Based on time-of-flight and known propagation speed of sound, the distance between the receiver and the transmitter can be calculated. The 3D position of the target is estimated via triangulation.	Accurate Inexpensive Lightweight	Line-of-sight is necessary The deployment is dense The speed of sound in air varies with environmental conditions.
Optical	Use various different types of vision sensors to observe the environment and detect trackers.	Provide plenty of information Fast Accurate Can work in a wide area	Line-of-sight is necessary Sensitive to illumination conditions and environmental noises

the inertial system. Although the method is straightforward, gyroscopes suffer from drifts caused by the zero rate offset. In the pose estimation approach proposed in this thesis, gravitational accelerations are used to estimation rotation angles. As no integration is involved, the approach avoids the common drift problem.

2.2.4 Summary of Sensing Technologies

Table 2.2 provides an overview of the different technologies, their advantages, and disadvantages.

2.3 Pose Estimation Strategy Overview

2.3.1 Notation, Coordinate Systems and Coordinate Transformations

Our fusion strategy manages inputs from infrastructure cameras and mobile sensors, therefore it is necessary to handle transformations between several coordinate systems. For the sake of clarity, the necessary definitions and notation used for each coordinate system are provided below, together with the list of variables used throughout the chapter. The coordinate transformation equations are also presented.

As schematically depicted in Fig. 2.6, four coordinate systems are involved:

- World coordinate system $\{w\}$: This is the global reference system used for describing the position and the orientation (pose) of objects; its z-axis points towards the sky, being perpendicular to the ground. All cameras have a known and fixed position, and orientation in this coordinate system.
- Accelerometer/object coordinate system $\{a\}$: Its orientation is aligned with the three accelerometer sensing axes. We assume that the object coordinate system has its origin in the center of the colored marker, and the marker is parallel to the x-axis of the accelerometer. For example, for a mobile device, this means in practice that while the marker can be put in a user-defined position, it has to be aligned with the device's border.
- Camera coordinate system $\{c\}$: For each camera, it is a Cartesian reference system attached to the camera, whose origin is located in the camera optical center. The z-axis is along the optical axis, and therefore x and y axes are parallel to the image plane.
- Camera frame coordinate system $\{f\}$: This 2D coordinate system is used to refer positions in the image plane in pixel units. The axes (to be called u-v) are parallel to the x-y axes of the camera coordinate system $\{c\}$. All calculations involving this reference frame are equivalent to those in the camera coordinate system but with a translation on the origin position, which instead of being in the image center is in the left-up corner of the image. So, this reference system will only be used for basic image processing, and then all positional data will be transformed to $\{c\}$ and exploited in that 3D reference system.

Let us denote $\{i\}$ and $\{j\}$ as two arbitrary coordinate systems (any of them may have the values $\{w\}$, $\{c\}$ or $\{a\}$). Table 2.3 summarizes the notation used in the following equations and in the rest of the chapter.

A 3D vector $\mathbf{v}$ is expressed as $\mathbf{v}_i$ in $\{i\}$, but expressed as $\mathbf{v}_j$ in $\{j\}$. The relationship between $\mathbf{v}_i$ and $\mathbf{v}_j$ can be expressed using a rotation matrix as:

$$\mathbf{v}_j = R_{ij}\mathbf{v}_i. \quad (2.1)$$

FIGURE 2.6: Coordinate systems involved in the proposed pose estimation system.

TABLE 2.3: Summary of variables in pose estimation.

Variables	Meanings
R_{ij}	3×3 rotation matrix from $\{i\}$ to $\{j\}$
$\mathbf{t}_{ij}$	3D translation vector from $\{i\}$ to $\{j\}$
ψ	Rotation angle about x-axis (pitch)
θ	Rotation angle about y-axis (roll)
ϕ	Rotation angle about z-axis(yaw)
g	Magnitude of gravity
D	The distance between two reference points
$\mathbf{P}_i = [X_i, Y_i, Z_i]^T$	A point in $\{i\}$, $i = w, c, a$
$\mathbf{P}_{hi} = [X_i, Y_i, Z_i, 1]^T$	A point in $\{i\}$, $i = w, c, a$, expressed by homogeneous coordinates
$\mathbf{P}_f = [u, v]^T$	A point in $\{f\}$
$\mathbf{P}_{hf} = [u, v, 1]^T$	A point in $\{f\}$, expressed by homogeneous coordinates
$\mathbf{g}_w = [0, 0, -g]^T$	Gravitational acceleration in $\{w\}$
$\mathbf{g}_a = [g_{ax}, g_{ay}, g_{az}]^T$	Gravitational acceleration in $\{a\}$
$\mathbf{s} = [x, y, z, \psi, \theta, \phi]^T$	6-dimensional pose vector which is composed by 3-dimensional position (in world coordinates) and 3 Euler angles.

In general, the transformation of a 3D position from a given reference frame to another

FIGURE 2.7: Object orientation expressed in pitch-roll-yaw rotations.

can be achieved by performing first a rotation between their reference frames and then a translation (related to the offset of the coordinate systems origins), which is mathematically expressed as:

$$\mathbf{P}_j = R_{ij}\mathbf{P}_i + \mathbf{t}_{ij}. \quad (2.2)$$

2.3.2 Body Orientation Representation

The body orientation has three independent degrees of freedom. Several representations can be used to express the orientation, for instance axis-angle, quaternions, and Euler angles (Diebel, 2006). They are equivalent and inter-convertible. In this work, we adopted the Euler angles representation for the object whose pose is to be estimated, which defines rotation angles taking into account the movement around three axes, as depicted in Fig. 2.7.

Each of those rotations have an associated elemental rotation matrix, which is defined as a function of the respective Euler angle, in next equations. A rotation of ψ radians about the x-axis (pitch) is defined as:

$$R_x(\psi) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos \psi & \sin \psi \\ 0 & -\sin \psi & \cos \psi \end{bmatrix}. \quad (2.3)$$

Similarly, a rotation of θ radians about the y-axis (roll) is defined as:

$$R_y(\theta) = \begin{bmatrix} \cos \theta & 0 & -\sin \theta \\ 0 & 1 & 0 \\ \sin \theta & 0 & \cos \theta \end{bmatrix}. \quad (2.4)$$

Finally, a rotation of ϕ radians about the z-axis (yaw) is defined as:

$$R_z(\phi) = \begin{bmatrix} \cos \phi & \sin \phi & 0 \\ -\sin \phi & \cos \phi & 0 \\ 0 & 0 & 1 \end{bmatrix}. \quad (2.5)$$

By composing the three elemental rotation matrices, the orientation of a frame with respect to a given reference frame can be expressed by a rotation matrix R . Depending on the order in which the roll, pitch, and yaw rotations are applied, in total, there are six possible orderings to compose R . As is explained in Pedley (2013), only two rotation sequences are capable to separate the yaw angle and solve the pitch and roll angles from accelerometer measurements, which are yaw-roll-pitch and yaw-pitch-roll. In this work, to obtain the rotation matrix from $\{w\}$ to $\{a\}$ (R_{wa}), three consecutive rotations might be performed in order: yaw – roll – pitch. Alternatively, yaw-roll-pitch can also be applied. From Eq. 2.3, Eq. 2.4, and Eq. 2.5, the complete rotation matrix may be obtained through multiplication:

$$R_{wa} = R_x(\psi)R_y(\theta)R_z(\phi) = \begin{bmatrix} \cos \theta \cos \phi & \cos \theta \sin \phi & -\sin \theta \\ \sin \psi \sin \theta \cos \phi - \cos \psi \sin \phi & \sin \psi \sin \theta \sin \phi + \cos \psi \cos \phi & \sin \psi \cos \theta \\ \cos \psi \sin \theta \cos \phi + \sin \psi \sin \phi & \cos \psi \sin \theta \sin \phi - \sin \psi \cos \phi & \cos \psi \cos \theta \end{bmatrix}. \quad (2.6)$$

An additional well-known property of rotation matrices is their orthogonality, i.e., they are invertible and the inverse, i.e., the rotation matrix of the opposite direction rotation, is equivalent to its transpose:

$$R_{ji} = R_{ij}^{-1} = R_{ij}^T. \quad (2.7)$$

2.3.3 Body Position Representation

In general, we have, for every point in the space, from Eq. 2.2:

$$\mathbf{P}_a = R_{wa}\mathbf{P}_w + \mathbf{t}_{wa}. \quad (2.8)$$

This equation can be evaluated at the position of the marker center (where $\mathbf{P}_a$ is zero), resulting for that point (and only for that point) in:

$$\mathbf{0} = R_{wa}\mathbf{P}_w + \mathbf{t}_{wa}. \quad (2.9)$$

Due to the definition of the translation vectors, this equation may be changed to:

$$\mathbf{0} = R_{wa}\mathbf{t}_{aw} + \mathbf{t}_{wa}. \quad (2.10)$$

FIGURE 2.8: Pose estimation system workflow.

In other words, the position of the object in $\{w\}$ ($\mathbf{t}_{aw}$) can be calculated as follows:

$$\mathbf{t}_{aw} = -R_{wa}^{-1}\mathbf{t}_{wa} = -R_{aw}\mathbf{t}_{wa}, \quad (2.11)$$

where the second equality comes from the invertibility of rotation matrices.

It is equivalent to the position of the world reference center in object coordinates, in opposite direction, and rotated back to world coordinates. This result will be exploited in the algorithm definitions.

2.3.4 System Description

An overview of the complete pose estimation system is given in Fig. 2.8. The sensor-based subsystem which contains a 3-axis accelerometer contributes to pitch and roll rotation angles estimation. With respect to pure vision-based approaches, the integration of the accelerometer measurements reduces the number of unknown pose parameters; therefore, robustness and computational efficiency are improved. Moreover, no acceleration integration process is involved. Therefore, we successfully avoid the drift problem and provide a zero-drift solution. Besides, the requirement of having an initial state is eliminated.

In the vision-based subsystem, the printable colored fiducial has to be attached to a visible location of the surface of the object, which has to be equipped with a 3-axis accelerometer. These requirements (enough visible surface and embedded accelerometer) are easily fulfilled by current smartphones and tablet devices. The proposed fiducial has a linear thin stripe-like geometry; it is thus different when compared against the conventional square

fiducials that are used in ARToolKit (*ARToolKit*) or ARTag (Fiala, 2005). Linear fiducials may better adapt to final services, because they are less invasive to the environment than the square fiducials, due to their smaller dimensions. The thinness also allows them to be attached to a small surface, for example, borders of mobile devices, hats or eyeglasses frames. Therefore, the proposed solution has the potential for indoor person tracking, robot tracking, mobile AR, and interaction in smart spaces.

Images from the vision sensors are processed to extract the position of the colored marker attached to the target device. The vision-based subsystem has several different forms depending on the number of cameras that detect the marker. It contributes to position and yaw estimation with/without the aid of accelerometers. For example, accelerometers are necessary in the monocular vision subsystem. Instead, they are not required in the stereo vision subsystem or the multi-camera subsystem. The system will be able to deliver the pose estimation when the target is within the coverage of at least one single camera. The accuracy of the estimation will differ (being presumably better when the marker is observed with more cameras), but the possibility of having a working system with a single camera may be enough for some services and make the whole system much more resilient to occlusions: even if the vision sensor deployment is done to have the space permanently covered by at least two sensors, the system will keep working almost seamlessly if, for any reason, in a given moment or period only a single sensor is available. The difference among the monocular vision, the stereo vision and the multi-camera approaches is related to the calculation of the markers' 3D position, to be described in detail in next sections.

2.4 Accelerometer as Inclination Sensor

Once the notation has been defined, how to extract the target device's inclination information from its accelerometers is the first issue to solve. Accelerometers sense both gravitational and dynamic (movement induced) accelerations. To measure inclination, gravitational accelerations need to be isolated, removing dynamic accelerations. To do so, a low-pass filter can be used. Some literature has done further research on this topic (Hees et al., 2013). In the case of Android mobile devices, a "gravity sensor" is embedded since API Level 9 (Android 2.3) was released. Thus we will assume that gravitational accelerations are perfectly isolated, focusing on the basic principle for converting the measurements of gravitational accelerations to inclination angles.

2.4.1 Pitch and Roll Estimation

In world reference coordinates the gravity vector ($\mathbf{g}_w$) is perfectly aligned with the z-axis, pointing in the negative direction. The 3-axis accelerometer provides the components of the gravitational acceleration expressed in the object reference frame ($\mathbf{g}_a = [g_{ax}, g_{ay}, g_{az}]^T$). Both gravity vectors are related through a rotation matrix, according to Eq. 2.1. For this case this

relation is:

$$\mathbf{g}_a = \begin{bmatrix} g_{ax} \\ g_{ay} \\ g_{az} \end{bmatrix} = R_{wa} \mathbf{g}_w = R_{wa} \begin{bmatrix} 0 \\ 0 \\ -g \end{bmatrix} = \begin{bmatrix} g \sin \theta \\ -g \sin \psi \cos \theta \\ -g \cos \psi \cos \theta \end{bmatrix}. \quad (2.12)$$

From Eq. 2.12, the pitch and roll angle can be finally deduced from the measurements of gravitational accelerations as follows:

$$\psi = \text{atan2}(g_{ay}, g_{az}), \quad (2.13)$$

$$\theta = \arcsin \frac{g_{ax}}{g}. \quad (2.14)$$

2.4.2 Accelerometer Calibration

Inertial sensors suffer from various error sources (Park and Gao, 2008). Most errors can be categorized into sensor bias, scale factor, axes misalignment, and noise. A simple model, convenient for short time intervals, of the measurement of the accelerometer in i -th axis ($a_i, i = x, y, z$) can be expressed:

$$\tilde{a}_i = a_i + S_i a_i + b_i + n_i, \quad (2.15)$$

where S_i is a scale factor, a_i is the actual (true) acceleration along i -th axis, b_i is the measurement bias, and n_i is the random noise.

In our system, the accelerometers are just used to measure the gravity. With uncalibrated accelerometers, gravity sensing will be biased, and therefore we will have errors in the pose estimation. In order to calibrate the three-axis accelerometer, the Six-Position Static Test method (Aggarwal et al., 2008) is used due to its simplicity and popularity. This method requires the accelerometer to be mounted on a leveled surface with each sensitive axis pointing alternatively up and down, which can be easily done in real settings. When the accelerometer points up, the measurement in the sensitive axis can be expressed as:

$$\tilde{a}_i^{up} = -(1 + S_i)g + b_i. \quad (2.16)$$

When the accelerometer points down, the measurement can be expressed as:

$$\tilde{a}_i^{down} = (1 + S_i)g + b_i. \quad (2.17)$$

Therefore, by summing and differencing Eq. 2.16 and Eq. 2.17, the bias and scale factors can be calculated as:

$$b_i = (\tilde{a}_i^{up} + \tilde{a}_i^{down})/2, \quad (2.18)$$

FIGURE 2.9: The plot of accelerometer measurements in z axis when z-axis points up.

$$S_i = (\tilde{a}_i^{down} - \tilde{a}_i^{up} - 2g)/2g. \quad (2.19)$$

We captured 500 samples in each of the six positions of the test. An example of the measurements in z axis when z-axis points up is plotted in Fig. 2.9.

The calibration parameters are listed in Table 2.4. Calculated bias and scale factors are used to correct the original measurements, so that in the rest of the system we use the calibrated (almost unbiased) gravity measurements (g_{ax}, g_{ay}, g_{az}):

$$\begin{aligned} g_{ax} &= (\tilde{g}_{ax} - b_x)/(1 + S_x), \\ g_{ay} &= (\tilde{g}_{ay} - b_y)/(1 + S_y), \\ g_{az} &= (\tilde{g}_{az} - b_z)/(1 + S_z), \end{aligned} \quad (2.20)$$

where $(\tilde{g}_{ax}, \tilde{g}_{ay}, \tilde{g}_{az})$ are the uncalibrated gravity measurements.

TABLE 2.4: Accelerometer calibration results.

	X axis	Y axis	Z axis
Bias	0.0002 m/s^2	-0.0001 m/s^2	-0.0011 m/s^2
Scale factor	0.0006	0.0006	0.0005

2.5 Camera as Position Sensor

Apart from accelerometer-based measurements, our system exploits images of the fiducial to derive the object's position. A key aspect of the system is the design of the fiducial marker, made to facilitate the extraction of the reference positions that are afterwards used as inputs

of the algorithms to calculate the device's position and orientation. In this section we will detail both the color and geometric features of the fiducial (two reference points) and the related fiducial detection algorithm based on ad-hoc image processing and computer vision procedures.

An additional key aspect to be described in this section is the geometric relation between the fiducial reference points and their projections in the camera frame coordinate system, as they are the other main input of our pose estimation algorithms apart from those from the accelerometer-based inclination measurements.

2.5.1 Fiducial Design

The fiducial is designed to: a) be able to provide two reference points referred to the target object; b) be easily recognizable within the environment, without causing confusion with other objects and resilient to illumination changes; c) take into consideration the compatibility and generalization to other applications; d) be low-cost. Considering these features, we propose a thin colored printable marker, which embodies three colored rectangles, as depicted in Fig. 2.10. The central part (in magenta) shares two edges with the lateral parts (in yellow and cyan), whose centers are treated as the reference points in our system (indicated by two crosses in Fig. 2.10). The whole marker is just several millimeters in width and the length could vary from a few centimeters to even longer, depending on the service working conditions. This linear feature makes it easy to be attached to object borders, e.g., the border of a mobile device. Thanks to its non-invasive characteristic with the environment, it can be considered for applications that are not compatible with obvious deployed markers in the environment.

FIGURE 2.10: The colored and shape-based fiducial used in the proposed system. The two black crosses are depicted to indicate the position of the two reference points, but they do not exist in the actual marker.

The colors used in our system are magenta, yellow, and cyan. However, they can be arbitrary combinations as long as they fulfill two principles. Firstly, the Hue range of selected colors should not overlap in Hue-Saturation-Value (HSV) color space. Secondly, they should be easily distinguishable from the typical colors in the surroundings (tracked object and environment).

Our simple approach for color discrimination of the fiducial in the image is to designate thresholds to each of the HSV components, separately for each color. The histograms in H, S, and V of the color “cyan” are shown in Fig. 2.11 under distinct illuminations. H values (the whole range is $[0, 180]$) almost cover the same narrow range under all the illuminations. S (the whole range is $[0, 255]$) and V (the whole range is $[0, 255]$) values vary and cover a wider range. To gain robustness against illumination changes, the adopted approach was to

aggregate the data under all the illuminations, as shown in Fig. 2.12 and define an interval between an upper and a lower threshold that achieves a 95% detection probability of the pixels in each of the fiducial segments, independently of illumination. The resulting thresholds for these three colors are listed in Table 2.5. In order to validate the performance of this thresholding, the selected thresholds were applied to an independent set of sample images. The detection procedure demonstrated to achieve an acceptable correct detection rate of the fiducial segments: 94%, 96%, and 94% under weak, adequate, and strong illuminations respectively.

TABLE 2.5: Selected thresholds for magenta, cyan, and yellow color segments in the fiducial.

	H	S	V
Yellow	15-40	5-160	91-255
Magenta	160-180	40-246	65-250
Cyan	95-115	11-239	68-243

2.5.2 Fiducial Detection Algorithm

After the fiducial design has been justified, let us describe the algorithm to detect the reference points from captured image sequences. The algorithm is designed taking into account both the color and the shape of the fiducial. Firstly, foreground is detected by frame differencing with a static background model (Piccardi, 2004), used to remove background (environment) areas with colors similar to those of the fiducial. Then, the three colors are segmented using the thresholds in Table 2.5 and the obtained regions of interest are converted to binary images. Note that the three colored rectangles in the fiducial are successive. Each of the detected regions for each of the fiducial colors is morphologically dilated using a kernel. Our experiments showed that a circular kernel with a radius of five pixels is adequate for this process, although system accuracy and robustness are not extremely dependent on this parameter. Afterwards, a logic “AND” operation is applied to the binary dilated images. The result of the operation are, for yellow-magenta and magenta-cyan combinations, overlapped regions approximately centered in the reference points.

The next step is to find the contours of these overlapped regions whose areas are not too big. Centroids of these contours are considered as reference point candidates. Let us assume that N reference point candidates are obtained. The two reference points are located at the two extremes of the central fiducial segment, so it is to decide which of the combination of 2 reference points out of N candidates is the relevant one. To do so, the algorithm calculates the percentage of “magenta” (central segment color) pixels in the segment between each pair of reference points. This percentage is compared with an upper threshold of $T = 90\%$, which has been experimentally validated as a suitable trade-off between the detection rate and imperfect color perception. Apart from the fiducial, it is rare to find regions composed by three selected colors in the foreground (at least in our environment). Based on this idea, the

FIGURE 2.11: The histograms in H, S, and V of the cyan color under distinct illuminations. The first column is under weak illumination. The second column is under adequate illumination. The third column is under strong illumination. (A-C) Examples. (D-F) Histograms in H. (G-I) Histograms in S. (J-L) Histograms in V.

algorithm simply chooses the longest candidate pair as the final result. In the stereo vision and multi-camera subsystems, further validation can be done by examining the distance between the final (reconstructed in 3D) reference points, given the known size of the fiducial. Algorithm 1 summarizes the previous processing.

An example of image processing is depicted in Fig. 2.13. We can observe that, in fact, single color segmentation covers a wide image portion; it is impossible to locate the fiducial

FIGURE 2.12: The histograms in H, S, and V of the cyan color under all three illuminations. (A) H. (B) S. (C) V.

Algorithm 1 Fiducial Detection

Input: Captured image I

Output: Reference points' positions in the image $(u^{(1)}, v^{(1)})$ and $(u^{(2)}, v^{(2)})$

- 1: Detect the foreground from the image I .
 - 2: Convert the image from the RGB color space to the HSV color space.
 - 3: Filter the image using the thresholds for each color and get three binary images, I_c (cyan), I_m (magenta), and I_y (yellow).
 - 4: Mask previous detections with foreground detection.
 - 5: Morphologically dilate I_c , I_m , and I_y separately using a disk kernel with a radius of five pixels.
 - 6: Do logic "AND" operation and get $I_m \& I_c = I_{mc}$, $I_m \& I_y = I_{my}$.
 - 7: Find contours of I_{mc} and I_{my} , then save centroids of each contour as candidates.
 - 8: Check the pixels between each two distinct candidates. If they are mostly magenta, that is if magenta pixels/all checked pixels $> T$, save the pair as one pair candidate.
 - 9: Among all the pair candidates, choose the longest pair as the final result.
-

directly on those images. However, the overlapped regions of two colors are just a few, so the search of suitable reference points is not computational intensive.

2.5.3 Projection of 3D Points into the Image Plane

A final aspect to tackle in this section is how 3D points in the scene are projected into the image plane. The pinhole camera model is typically considered, which meets the collinear condition, i.e., the world point, the camera projection center, and the projected point are collinear (Hartley and Zisserman, 2003), as illustrated in Fig. 2.14.

The projection of a 3D position (for instance, a fiducial reference point) in world coordinates $\{w\}$ ($\mathbf{P}_w$) into a 2D image is done in two steps. Firstly, $\mathbf{P}_w$ is transformed to $\{c\}$, resulting as $\mathbf{P}_c$, using R_{wc} and $\mathbf{t}_{wc}$ parameters as follows:

$$\mathbf{P}_c = R_{wc}\mathbf{P}_w + \mathbf{t}_{wc}. \quad (2.21)$$

FIGURE 2.13: An example of image processing. (A) Original RGB image. (B) Image represented in HSV color space. (C) Filtered yellow region. (D) Filtered magenta region. (E) Filtered cyan region. (F) Overlaps of yellow and magenta regions. (G) Overlaps of cyan and magenta regions.

FIGURE 2.14: The pinhole camera model.

Using an equivalent homogeneous representation of $\mathbf{P}_w$ as a 4D vector ($\mathbf{P}_{hw}$ according to our notation), defined as:

$$\mathbf{P}_{hw} = \begin{bmatrix} \mathbf{P}_w \\ 1 \end{bmatrix}. \quad (2.22)$$

we may rewrite Eq. 2.21 as:

$$\mathbf{P}_c = [R_{wc} \ \mathbf{t}_{wc}] \mathbf{P}_{hw}. \quad (2.23)$$

FIGURE 2.15: Camera calibration parameters.

The rotation matrix R_{wc} and the translation vector $\mathbf{t}_{wc}$ are called *extrinsic parameters*, as it indicates a rigid 3D to 3D transformation from the world coordinate system to the camera coordinate system, as illustrated in Fig. 2.15.

Then $\mathbf{P}_c$ is projected to the image plane (expressed in homogeneous format) as $\mathbf{P}_{hf}$, whose coordinates are given in pixels by the equation:

$$\lambda \mathbf{P}_{hf} = K \mathbf{P}_c = \begin{bmatrix} f_x & 0 & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \mathbf{P}_c, \quad (2.24)$$

where λ is a scale factor and elements of K are camera intrinsic parameters: the focal length (f_x, f_y) and the principal point (u_0, v_0).

This step is a 3D to 2D perspective projection which transforms 3D camera coordinates to 2D image coordinates. The focal length and the principal point are called *intrinsic parameters*, which are determined by the camera itself, as illustrated in Fig. 2.15.

Finally, the concatenation of the two transformations, from world coordinates (in homogeneous format, $\mathbf{P}_{hw}$) to camera frame coordinates (also in homogeneous format, $\mathbf{P}_{hf}$) results in:

$$\lambda \mathbf{P}_{hf} = M \mathbf{P}_{hw}, \quad (2.25)$$

where M is a 3×4 projection matrix summarizing the whole projection process:

$$M = K[R_{wc} \ \mathbf{t}_{wc}]. \quad (2.26)$$

Camera calibration, also referred to as *camera resectioning*, is the process of estimating the parameters of a pinhole camera model to approximate the projective mapping from world coordinates to pixel coordinates. The parameters include intrinsic parameters (K) and extrinsic parameters (R_{wc} and $\mathbf{t}_{wc}$), or equivalently the M matrix for a given spatial scenario and camera deployment. In our deployment, the calibration was done offline, using the Matlab[®] Calibration Toolbox, which implements the method proposed by Zhang (1999), targeted to minimize the total re-projection error.

From the previous description, it is clear that projected fiducial reference points in the image plane contain information on the 3D position of those points in space. From only one of those projected points it is impossible to obtain the complete 3D position of the point, but fusion of several cameras, and/or additionally exploitation of inclination information from accelerometers may result in estimation of the whole object pose. Describing the procedures to do so is the objective of the following section.

2.6 Pose Estimation Using Fused Vision and Inertial Data

In this section, we will present different fusion algorithms combining previously described acceleration data and input from vision sensors. The difference among these fusion algorithms is the 3D reconstruction of the two fiducial reference points. Depending on the number of cameras that have the marker in view, we propose different fusion algorithms. First of all, we will describe the data synchronization procedure in Section 2.6.1. Then, we will explain in Section 2.6.2 the monocular vision object positioning system to deal with the situation when the reference point is seen by a single camera. Section 2.6.3 will describe the stereo vision object position extraction, while Section 2.6.4 will explain the multi-camera triangulation algorithm. The algorithm proposed in Section 2.6.5 deals with the situation when the reference point is partially occluded. Once 3D positions of those points in world coordinates are obtained by either method, the complete estimation of the 6-DoF pose is performed. The common procedure for this derivation is described in Section 2.6.6.

2.6.1 Data Synchronization

Synchronization among different sensors is required and important in hybrid tracking systems, which is difficult to achieve in practice (Jiang, Neumann, and You, 2004). In the proposed system, cameras are set to run at about 25 frames/s. Captured images are transmitted to the server either by wireless connection or by cable depending on the types of cameras. Accelerometers run in the client side at around 100 Hz. The communication between the server and the client is designed as wireless via TCP/IP, running in a separate thread. The adopted approximate synchronization approach for real-time operation is described as follows. At the beginning of each loop, the server starts grabbing visual data from cameras and simultaneously sends a data request message to the client. Then, the client sends back the current accelerometer measurements when the message is received. One thread in the server listens to the client and another thread processes the images to locate the marker. Finally, marker detection results are integrated with accelerometer measurements to recover the six-DoF pose. The temporal description of each loop is illustrated in Fig. 2.16. It is worth mentioning that there is a latency in accelerometer measurements due to the transmission of the data request message via the wireless communication link (around 3-5 ms for a single

FIGURE 2.16: The temporal description of each loop.

trip in the current network). However, it is acceptable considering the fast update rate of the accelerometer measurements.

An alternative solution is to synchronize the clocks on the server and the client by measuring the clock offset using the IEEE 1588 precision time protocol (Kannisto et al., 2005). Once synchronized, the data request message sent to the client is enclosed by the time stamp from the server. Then, the client calculates the local corresponding time stamp at which the message is sent and searches backward for the accelerometer measurements at that time stamp. In this approach, a buffer is necessary to save the accelerometer measurements and the time stamps, whose size should be big to save data during the frame update.

2.6.2 Monocular Vision Positioning System

In the case the fiducial is in the coverage of just one camera or it is partially occluded for some cameras, just being visible by one of them, a monocular positioning system may be used. The process here is to define a set of constraints on fiducial reference points' positions leading to a potential solution. The process here needs to jointly estimate the positions of both fiducial reference points, $\mathbf{P}_w^{(1)}$ and $\mathbf{P}_w^{(2)}$, expressed in world coordinates, so we need to estimate 6 unknown variables (as both $\mathbf{P}_w^{(1)}$ and $\mathbf{P}_w^{(2)}$ have three dimensions).

In our vision sensor modeling we mentioned the collinear condition in the pinhole camera model, which also holds for the monocular vision subsystem. Considering a 2D point in the image, there exists a collection of 3D points that are mapped onto the same point. These points lay on the ray connecting the camera projection center and the 2D point, as depicted in Fig. 2.17a for the two fiducial reference points. From that figure, let $\mathbf{P}_f^{(1)}$ and $\mathbf{P}_f^{(2)}$ be the projection of $\mathbf{P}_w^{(1)}$ and $\mathbf{P}_w^{(2)}$ in $\{f\}$, respectively. Rearranging Eq. 2.25, all potential 3D points lying in the ray associated to a pixel point can be mathematically expressed as:

$$[X_w^{(i)}, Y_w^{(i)}, Z_w^{(i)}]^T = -R_{wc}^{-1} \mathbf{t}_{wc} + \lambda^{(i)} R_{wc}^{-1} K^{-1} [u^{(i)}, v^{(i)}, 1]^T, i = 1, 2, \quad (2.27)$$

FIGURE 2.17: Monocular vision object positioning system. (A) System geometry. $P_f^{(1)}$ and $P_f^{(2)}$ are the image plane projections of the two respective fiducial reference points, $P_w^{(1)}$ and $P_w^{(2)}$. (B) System workflow.

where $\lambda^{(i)}$ has the same meaning as in Eq. 2.24 for the i -th fiducial reference point, defining the position of the 3D point along the ray.

Each of the reference points has an associated **collinearity constraint**: reference points must fulfill this equation for a given (unknown) value of $\lambda^{(i)}$. So, those collinearity constraints convert the problem of solving 6 unknown variables to a reduced problem of solving two unknown variables ($\lambda^{(1)}$ and $\lambda^{(2)}$ values in the parametric formulation of the projection lines). A simpler form of this equation is:

$$[X_w^{(i)}, Y_w^{(i)}, Z_w^{(i)}]^T = \mathbf{t}_{cw} + \lambda^{(i)} [X_t^{(i)}, Y_t^{(i)}, Z_t^{(i)}]^T, i = 1, 2, \quad (2.28)$$

where we define an auxiliary variable $[X_t^{(i)}, Y_t^{(i)}, Z_t^{(i)}]^T$ as follows:

$$[X_t^{(i)}, Y_t^{(i)}, Z_t^{(i)}]^T = R_{wc}^{-1} K^{-1} [u^{(i)}, v^{(i)}, 1]^T. \quad (2.29)$$

We can also define a **distance constraint** for the fiducial reference points. The distance between the two reference points in space is known (D), which allows defining the relation:

$$\|\mathbf{P}_w^{(1)} - \mathbf{P}_w^{(2)}\| = D. \quad (2.30)$$

We just need one more constraint on reference points' positions, to be able to solve the problem. The additional constraint we are including is an **inclination constraint**, defined as follows. The positions of the reference points in the accelerometer/object coordinates are $\mathbf{P}_a^{(1)} = [-0.5D, 0, 0]^T$ and $\mathbf{P}_a^{(2)} = [0.5D, 0, 0]^T$ (we assume that the fiducial marker is centered at the origin of the accelerometer/object coordinate system and aligned with its x-axis). The

difference vector between those two points in these coordinates is:

$$\Delta \mathbf{P}_a = \mathbf{P}_a^{(2)} - \mathbf{P}_a^{(1)} = [D, 0, 0]^T. \quad (2.31)$$

We may rotate this difference into world coordinates, resulting in:

$$\Delta \mathbf{P}_w = \mathbf{P}_w^{(2)} - \mathbf{P}_w^{(1)} = R_{wa}^{-1} [D, 0, 0]^T. \quad (2.32)$$

From Eq. 2.32 and Eq. 2.6 we get:

$$\Delta \mathbf{P}_w = D [\cos \theta \cos \phi, \cos \theta \sin \phi, -\sin \theta]^T. \quad (2.33)$$

This relation imposes three constraints (one per Cartesian coordinate) on the fiducial reference points' relative positions in world coordinates, involving two Euler angles. Especially relevant for us is the z-axis related **inclination constraint**:

$$\Delta Z_w = Z_w^{(2)} - Z_w^{(1)} = -D \sin \theta. \quad (2.34)$$

Since the accelerometers provide an independent measurement of the inclination (roll) angle, as described in Eq. 2.14, we may derive an equivalent inclination constraint as:

$$Z_w^{(2)} - Z_w^{(1)} = -D g_{ax} / g. \quad (2.35)$$

This idea allows a reformulation of the inclination constraint, building a bridge between the accelerometer measurements and the locations of the two reference points. Combining Eq. 2.35 and Eq. 2.28, we obtain the following relation between $\lambda^{(1)}$ and $\lambda^{(2)}$ parameters:

$$\lambda^{(1)} = \frac{D g_{ax}}{Z_t^{(1)} g} + \frac{\lambda^{(2)} Z_t^{(2)}}{Z_t^{(1)}}. \quad (2.36)$$

We may substitute $\lambda^{(1)}$ in Eq. 2.30 and rearrange it, obtaining a quadratic equation:

$$a \lambda^{(2)^2} + b \lambda^{(2)} + c = 0, \quad (2.37)$$

where

$$\begin{aligned} a &= \left(\frac{Z_t^{(2)}}{Z_t^{(1)}} Y_t^{(1)} - Y_t^{(2)} \right)^2 + \left(\frac{Z_t^{(2)}}{Z_t^{(1)}} X_t^{(1)} - X_t^{(2)} \right)^2, \\ b &= \frac{2 D g_{ax}}{Z_t^{(1)^2} g} [Z_t^{(2)} (X_t^{(1)^2} + Y_t^{(1)^2}) - Z_t^{(1)} (X_t^{(1)} X_t^{(2)} + Y_t^{(1)} Y_t^{(2)})], \\ c &= D^2 \left[\left(\frac{g_{ax}}{g Z_t^{(1)}} \right)^2 (X_t^{(1)^2} + Y_t^{(1)^2} + Z_t^{(1)^2}) - 1 \right]. \end{aligned}$$

This equation yields two solutions, but only one (the positive) is physically feasible, because the positive value means the point lays in front of the camera. Once we get the value of $\lambda^{(2)}$,

FIGURE 2.18: Mono-view object positioning system geometry.

$\lambda^{(1)}$ can be obtained using Eq. 2.36, and 3D positions of both reference points are easily calculated with Eq. 2.28.

In order to reduce the wrong detection rate of the marker, we introduced the Algorithm 1 to detect the two marker points in one image. Through checking the percentage of the pixels between two marker points candidates in color magenta, the detection rate is maintained high. However, these two marker points can also be detected separately, leading to noisy results. Alternatively, the paper marker can be replaced by two LEDs, as is done in (Li et al., 2015b). In this case, the monocular vision solution can also be expanded to solve the problem when each reference point is seen by one camera, as illustrated in Fig. 2.18.

The process is similar to the aforementioned monocular vision solution. The only issue that needs to be paid attention is to use corresponding camera parameters in Eq. 2.28, that is, R_{wc} , t_{wc} , and K .

2.6.3 Stereo Vision Positioning System

In case that two cameras detect the fiducial simultaneously, 3D positions of each fiducial reference point can be obtained by triangulation, as depicted in Fig. 2.19a. The process described below is an adaptation of the linear least-squares method (Nocedal and Wright, 2006). Other triangulation procedures (Hartley and Sturm, 1997) might be used to solve this part of the problem.

To convert this geometry to algebraic expressions, we let M_L and M_R denote the calibrated world to image plane projection matrices of left and right cameras respectively, and let λ_L and λ_R denote the respective multipliers. Let $[u_L, v_L]$ and $[u_R, v_R]$ be the 2D projections of one reference point ($\mathbf{P}_w = [X_w, Y_w, Z_w]^T$) in the left and right image plane. Applying

FIGURE 2.19: Stereo vision object positioning system. (A) System geometry. P_w is the position of one fiducial reference point, to be obtained through triangulation. (B) System workflow.

Eq. 2.24 to each camera, we obtain the following equations for each reference point:

$$\lambda_L[u_L, v_L, 1]^T = M_L[X_w, Y_w, Z_w, 1]^T, \quad (2.38)$$

$$\lambda_R[u_R, v_R, 1]^T = M_R[X_w, Y_w, Z_w, 1]^T. \quad (2.39)$$

Rearranging Eq. 2.38 and Eq. 2.39, the following over-determined linear equations system can be obtained:

$$A\mathbf{P}_w = b, \quad (2.40)$$

where A is a 4×3 matrix and b is a 4×1 matrix, described next:

$$A = \begin{bmatrix} M_L(1,1) - u_L M_L(3,1) & M_L(1,2) - u_L M_L(3,2) & M_L(1,3) - u_L M_L(3,3) \\ M_L(2,1) - v_L M_L(3,1) & M_L(2,2) - v_L M_L(3,2) & M_L(2,3) - v_L M_L(3,3) \\ M_R(1,1) - u_R M_R(3,1) & M_R(1,2) - u_R M_R(3,2) & M_R(1,3) - u_R M_R(3,3) \\ M_R(2,1) - v_R M_R(3,1) & M_R(2,2) - v_R M_R(3,2) & M_R(2,3) - v_R M_R(3,3) \end{bmatrix}, \quad (2.41)$$

$$b = \begin{bmatrix} u_L M_L(3,4) - M_L(1,4) \\ v_L M_L(3,4) - M_L(2,4) \\ u_R M_R(3,4) - M_R(1,4) \\ v_R M_R(3,4) - M_R(2,4) \end{bmatrix}.$$

To estimate the position of the fiducial reference point we may solve the equations using a least squares approach minimizing $\|A\mathbf{P}_w - b\|$. Then we will get the reference point coordinates in $\{w\}$ as:

$$\mathbf{P}_w = A^+ b, \quad (2.42)$$

where A^+ is the pseudo-inverse matrix of A .

The same process is done for the 3D reconstruction of the other fiducial reference point. This process provides the positions of both fiducial reference points expressed in world coordinates, to be called $\mathbf{P}_w^{(1)}$ and $\mathbf{P}_w^{(2)}$.

2.6.4 Multi-camera Solution by Triangulation

When a point is viewed by N ($N > 2$) cameras, several approaches can be used for multi-camera triangulation. One solution is the triangulation based on the least squares minimization as is done in the stereo vision system. Each pair of 3D-2D corresponding points in each camera meets the relation:

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = M \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} = \begin{bmatrix} M(1,1) & M(1,2) & M(1,3) & M(1,4) \\ M(2,1) & M(2,2) & M(2,3) & M(2,4) \\ M(3,1) & M(3,2) & M(3,3) & M(3,4) \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix}. \quad (2.43)$$

Eliminating λ leads to two linear equations:

$$\begin{aligned} (M(3,1)X_w + M(3,2)Y_w + M(3,3)Z_w + M(3,4))u \\ = M(1,1)X_w + M(1,2)Y_w + M(1,3)Z_w + M(1,4), \\ (M(3,1)X_w + M(3,2)Y_w + M(3,3)Z_w + M(3,4))v \\ = M(2,1)X_w + M(2,2)Y_w + M(2,3)Z_w + M(2,4). \end{aligned} \quad (2.44)$$

When the marker is viewed by N cameras, the following over-determined linear equations system can be obtained:

$$A\mathbf{P}_w = b, \quad (2.45)$$

where A is a $2N \times 3$ matrix and b is a $2N \times 1$ matrix, described next:

$$\begin{aligned} A &= \begin{bmatrix} M_1(1,1) - u_1M_1(3,1) & M_1(1,2) - u_1M_1(3,2) & M_1(1,3) - u_1M_1(3,3) \\ M_1(2,1) - v_1M_1(3,1) & M_1(2,2) - v_1M_1(3,2) & M_1(2,3) - v_1M_1(3,3) \\ \vdots & \vdots & \vdots \\ M_N(1,1) - u_NM_N(3,1) & M_N(1,2) - u_NM_N(3,2) & M_N(1,3) - u_NM_N(3,3) \\ M_N(2,1) - v_NM_N(3,1) & M_N(2,2) - v_NM_N(3,2) & M_N(2,3) - v_NM_N(3,3) \end{bmatrix}, \\ b &= \begin{bmatrix} u_1M_1(3,4) - M_1(1,4) \\ v_1M_1(3,4) - M_1(2,4) \\ \vdots \\ u_NM_N(3,4) - M_N(1,4) \\ v_NM_N(3,4) - M_N(2,4) \end{bmatrix}. \end{aligned} \quad (2.46)$$

FIGURE 2.20: The geometry of the occlusion handling system.

Afterwards, we do the same as in the stereo vision system to solve the equations using a least squares approach minimizing $\|A\mathbf{P}_w - b\|$ and get

$$\mathbf{P}_w = A^+b. \quad (2.47)$$

Another alternative is to apply the stereo-view solution for each combination of two cameras. There will be C_N^2 solutions. Then, the median of the all the solutions is considered as the final result.

2.6.5 Constraint-based Occlusion Handling

In this section, we propose a new fusion algorithm to determine the 6-DoF pose in case of partial occlusions, which refers to the situations when one reference point is located in the view of one single camera. Let us start with the case when the right reference point is seen by one camera and the left reference point is seen by two or more cameras. The 3D position of the left reference point $\mathbf{P}_L$ is obtained by triangulation as is explained in the previous section. The geometry of the occlusion handling system is depicted in Fig. 2.20.

As we explained before, Eq. 2.35 imposes an inclination constraint on the Z elements of the two reference points. Thus, the element Z_R of $\mathbf{P}_R$ can be calculated as

$$Z_R = Z_L - Dg_x/g. \quad (2.48)$$

According to the camera projection model, the relationship between the reference point's 3D position $\mathbf{P}_R$ in $\{w\}$ and its 2D position (u_R, v_R) in the image can be expressed as

$$\lambda \begin{bmatrix} u_R \\ v_R \\ 1 \end{bmatrix} = M \begin{bmatrix} X_R \\ Y_R \\ Z_R \\ 1 \end{bmatrix}, \quad (2.49)$$

where λ is a scale factor and M is a 3×4 transformation matrix.

Therefore, in the system of equations Eq. 2.49, there are three unknown parameters (X_R , Y_R and λ) and three equations. Thus, the 3D position of the right reference point can be calculated by this determined system. The same process is done in case that the left reference point is partially occluded.

2.6.6 Complete 6-DoF Pose Estimation

The monocular, stereo vision and the multi-camera system as well as the occlusion handling solution provide the positions of both fiducial reference points expressed in world coordinates $(\mathbf{P}_w^{(1)}, \mathbf{P}_w^{(2)})$, and at the same time accelerometer sensors are able to give pitch and roll estimation, as shown in Section 2.4.1 (Eq. 2.13 and Eq. 2.14).

To obtain a complete 6-DoF pose estimation we may perform the following calculations. Due to the symmetrical design of the fiducial, the position of its central point is considered as the position of the object. So we may estimate the object's position in world coordinates as the average position of both reference points in world coordinates:

$$\mathbf{t}_{aw} = (\mathbf{P}_w^{(1)} + \mathbf{P}_w^{(2)})/2. \quad (2.50)$$

To obtain the remaining Euler angle (yaw) we may exploit the remaining relations in Eq. 2.32. The differences of the positions of the reference points in x and y axes in world coordinates, according to Eq. 2.32 are:

$$\Delta X_w = X_w^{(2)} - X_w^{(1)} = D \cos \theta \cos \phi, \quad (2.51)$$

$$\Delta Y_w = Y_w^{(2)} - Y_w^{(1)} = D \cos \theta \sin \phi. \quad (2.52)$$

So, dividing both equations we obtain:

$$\phi = \text{atan2}(Y_w^{(2)} - Y_w^{(1)}, X_w^{(2)} - X_w^{(1)}). \quad (2.53)$$

Summarizing, the 6-DoF object pose estimate results, from gravity measures and fiducial reference positions:

$$\mathbf{s} = \begin{bmatrix} x \\ y \\ z \\ \psi \\ \theta \\ \phi \end{bmatrix} = \begin{bmatrix} (X_w^{(1)} + X_w^{(2)})/2 \\ (Y_w^{(1)} + Y_w^{(2)})/2 \\ (Z_w^{(1)} + Z_w^{(2)})/2 \\ \text{atan2}(g_{ay}, g_{az}), \\ \arcsin(g_{ax}/g) \\ \text{atan2}(Y_w^{(2)} - Y_w^{(1)}, X_w^{(2)} - X_w^{(1)}) \end{bmatrix}. \quad (2.54)$$

2.7 System Evaluation

In this section we will experimentally assess the proposed pose estimation system's accuracy and calculate the computational load. Firstly, Section 2.7.1 describes the setup for the experiments. Section 2.7.2 explains the approaches that we have used to evaluate the system performance. Section 2.7.3 provides the experimental results for the accuracy assessment. Additionally, the computational load of both solutions in a currently commercial workstation (HP Z420) is featured in Section 2.7.4. Finally, we discuss the experimental results in Section 2.7.5.

2.7.1 Implementation and Prototype Setup

The prototype for validation involves two web cameras (Logitech C920) and an Android mobile device (Google Nexus 10). The cameras are calibrated with Matlab camera calibration toolbox using the method proposed by Zhang (1999) and deployed in a way that they cover as much as possible of the moving object space. The marker is composed of a yellow rectangle (3 cm×1 cm), a magenta rectangle (15 cm×1 cm), and a cyan rectangle (3 cm×1 cm), from left to right. The marker is pasted on the top edge or in the top back of the mobile device, in order to facilitate its visibility from the camera network when the user is pointing at the environment as s/he would be using an indoor AR service.

In the case of Android mobile devices, the accelerometers are integrated in the device and since API Level 9 (Android 2.3), the gravitational acceleration can be directly obtained from the gravity sensor. The accelerometers are calibrated using the Six-Position Static Test method as explained in Section 2.4.2.

The server side is coded with C++ and the client side, i.e., the mobile device is based on Android programming. For image processing, some functions from OpenCV libraries are used.

2.7.2 Testing Procedure

Two methodologies are proposed to evaluate the system accuracy. The first comparison is against a set of real registered positions. The second experiment benchmarks the system with respect to an off-the-shelf tool (OpenCV library) providing accurate marker-based position and orientation estimates.

2.7.2.1 Comparison with Real Measurements to Evaluate Accuracy

This experiment is designed to provide a quantitative evaluation of the system's performance in terms of accuracy. Intuitively speaking, the accuracy of each pose varies depending on the distance to the cameras and the angle between the marker and the cameras. Accuracy is tested against real positions in two dimensions, which are the distance between the marker and the cameras, and the inclination of the mobile device. The real measurements are obtained through a Laser Distance Meter with an accuracy of few millimeters in terms of position, and a goniometer with the resolution of 0.1 degree. In this test, a comparison is made between the results given by the proposed system and the real measurements in 6-DoF, that is, the position estimation errors in each of the 3 axes separately (in millimeters) and the rotation angles estimation errors around 3 axes separately (in degrees).

2.7.2.2 Comparison with OpenCV Grid Finder to Evaluate the Displacement in the Image Plane

The second experiment compares the system performance against the accurate marker-based camera pose estimation method by OpenCV, an open source, widely-used, and cross-platform computer vision library. OpenCV finds the position of internal corners of a chessboard using the function *findChessboardCorners()* and then calculates an object's pose from 3D-2D point correspondences using the function *solvePnP()*. The difference with the proposed system is that OpenCV uses the internal camera of the mobile device to take the grid-fiducial view, while the proposed system performs the recognition of the marker through two external cameras and the device's accelerometer. Thus the proposed system uses external cameras for pose estimation and embedded camera for augmented reality displays, while OpenCV Grid Finder is using the same camera for camera pose estimation and display. The different output between the two systems is compared in terms of displacement in the image plane between the virtual objects and the real objects. This validation approach makes feasible to test the performance of the proposed system for augmented reality-like applications.

To set this benchmark up, a chessboard composed of 6×8 grids of $30 \text{ mm} \times 30 \text{ mm}$ has been chosen as fiducial (Fig. 2.21). The physical augmented object is a 4×4 grid (forming 25 corners) with total dimension of $360 \text{ mm} \times 280 \text{ mm}$. The 25 corners (with known positions in

FIGURE 2.21: Sample images showing the results of the two pose estimation systems. The green and red crosses correspond to the projections of the 25 test points using the results from the proposed and the reference systems, respectively. The 6×8 , 30×30 mm chessboard pattern is used by the reference system to estimate the camera 3D pose.

global coordinate system) are projected to the image plane using the transformation matrices calculated by these two tools. Finally, the displacements (in pixels) between the projected points and the real points in the image plane are calculated.

In this experiment, for simplicity's sake, three distances have been defined. The first two are the measurement distances, which refer to: a) the distance between the tablet and the chessboard in OpenCV Grid Finder and to b) the distance between the external cameras and the tablet in the proposed system. The third distance is the augmentation distance, which is between the physical object and the tablet camera. To make the result comparable, the measurement distances are kept the same in the two approaches.

2.7.3 Accuracy Assessment

2.7.3.1 Accuracy: Benchmark to Real Measurements

We have implemented two experiments with different camera setup to assess the accuracy of the system. In the first experiment, the camera's frame rate has been set to 10 fps and the resolution of the frame is 864×480 pixels. Tests were performed in 16 poses changing the distance between the tablet and the cameras (50, 150, 250, and 350 cm) along the depth direction and the inclination of the tablet (0° , 45° , 90° , and 180°). In each experiment, the estimated position and orientation (in the 3 axes) was compared with the measurements from Laser Distance Meter and goniometer. In each pose, the estimation results corresponding to

TABLE 2.6: Error in translation and orientation estimation from different distances and different orientations. The average estimation error at each distance is also shown. The X-axis is in the horizontal plane pointing to the right. The Y axis is the depth direction and the Z axis points upwards.

Distance (cm)	Position error (mm)			Orientation error (degrees)		
	X axis	Y axis	Z axis	X axis	Y axis	Z axis
50	3.2	6.5	2.4	0.6	1.5	1.8
150	9.1	9.1	3.1	0.8	0.4	1.6
250	11.5	39.2	7.4	1.3	1.1	3.7
350	15.3	87.1	6.1	1.5	1.0	3.0
Average	9.8	37.4	4.7	1.0	1.0	2.6

the proposed system are given as average of 600 samples. The errors are shown in Table 2.6 and Fig. 2.22. Table 2.6 contains the averaged error for all the inclinations in each distance (20-minute data).

As it can be inferred from Table 2.6 and Fig. 2.22, the position estimation error increases with the distance between the mobile device and the cameras. The most significant errors occur in the y axis, which is almost parallel to the optical axes of the cameras. This is expected because stereo vision systems are not very sensitive to the depth change, as the depth information is lost during the projection from 3D to 2D. Stereo vision perceives the depth information by triangulation, thus a small error in the 2D image point may cause a significant error in the depth estimation. Additionally, the orientation error is larger in the z axis which is determined by the reconstructed two marker points. As the marker goes further from the cameras, the error is also enlarged during 3D reconstruction. What is more, the gravitational accelerometer is noisy and not perfectly isolated from the acceleration.

In the second experiment, we have implemented it with different camera setup and assessed the accuracy of both the stereo vision system and the monocular vision system. We have used two Logitech HD Pro webcam C920 cameras. They are deployed in the ceiling of our laboratory to have a bird view of the scene, as depicted in Fig. 2.23a. Examples of captured images from the left and right camera are shown in Fig. 2.23b and Fig. 2.23c. The resolution of the image is set to 640×360 pixels and the cameras run at 20 frames/s. The fiducial, whose size is 19×1 cm, is tagged on the upper border of a Nexus 10 tablet, which is held in landscape mode. The image processing and pose estimation process are performed in real time in an HP Z420 workstation with an Intel® Xeon® CPU E5-1620 @ 3.60GHz. Cameras and accelerometer are calibrated offline in advance.

On this infrastructure, we have performed static accuracy measurements in seven positions, centered between both cameras, and at increasing distance from the wall next to both cameras, as indicated by the orange line in Fig. 2.23a. The distance to the wall varies from 113 cm to 413 cm and tests have been performed every 50 cm. The tablet is placed in a static stand at a predefined pitch angle (50 degrees), measured by a goniometer with 0.1 degree resolution. The distance has been measured using a Laser Distance Meter with high accuracy. At

FIGURE 2.22: The average estimation errors for different distances between the mobile device and the external cameras. (A) Position estimation errors. (B) Rotation estimation errors.

each position, 50 images from left and right cameras have been captured and corresponding accelerometer measurements have been transmitted to the central workstation, which processes the images and locates the reference points in the captured images. Then, we have compared the estimated position and rotation accuracy with the ground truth and the results of the root-mean-square (RMS) errors are shown in Table 2.7.

As is indicated in Table 2.7, regarding the position estimation, the stereo vision system provides higher accuracy than the monocular vision system. This is expected as the stereo vision system optimizes the position estimation by triangulation. However, the monocular vision system reduces the requirement for the density of the camera deployment. Both systems provide high accuracy in orientation estimation, as two rotation angles are calculated from accelerometer measurements regardless of the number of cameras that see the marker.

FIGURE 2.23: Experimental setup. (A) Deployment of cameras in our lab. (B) A captured image from the left camera. (C) A captured image from the right camera.

TABLE 2.7: RMS of estimated position and rotation errors in the stereo vision system and the monocular vision system

Distance (cm)	Stereo vision system		Monocular vision system	
	Position error (mm)	Orientation error (degree)	Position error (mm)	Orientation error (degree)
113	8.8	0.4	48.3	1.4
163	2.6	1.3	74.3	0.7
213	13.2	0.7	79.4	2.6
263	15.2	1.7	112.2	1.7
313	17.1	3.1	212.6	1.6
363	19.0	2.7	136.1	2.3
413	28.9	4.2	275.4	2.6
Average	16.8	2.3	155.4	2.0

2.7.3.2 Visible Displacement: Benchmark to OpenCV Grid Finder

This experiment is focused on the comparison against the performance of the OpenCV Grid Finder which, as previously said, it is a tool that enables the deployment of AR fiducial-based applications. Table 2.8 shows the results for different measurement distances and augmentation distances. An example of the results of the two pose estimation systems is shown in Fig. 2.21.

Both systems have an average error around 10 pixels. Additionally, the proposed system has lower variance than the reference system, thus gaining in stability. At 1 meter distance, the proposed system average is 9.01 pixels, while the OpenCV Grid Finder provides 11.52 pixels. At 2 meters, the error is 13.67 and 10.4 pixels, respectively. An important feature is that, when the mobile device is 3 meters away from the webcams and the chessboard,

TABLE 2.8: Mean errors and variance of the two considered pose estimation systems.

Measurement distance (cm)	Augmentation distance (cm)	Proposed system		Marker-based reference system	
		Mean error (pixel)	Variance	Mean error (pixel)	Variance
100	50	7.4	2.7	19.7	7.0
100	150	4.0	0.6	8.9	0.6
100	250	15.2	0.3	12.4	0.4
100	350	9.7	0.4	5.1	0.3
200	50	18.3	5.4	17.4	31.1
200	150	12.9	1.5	2.0	0.7
200	250	12.0	0.3	7.2	0.4
200	350	11.6	0.3	9.9	0.5
300	50	10.0	4.8	N/A	N/A
300	150	17.6	6.5	N/A	N/A
300	250	15.3	1.0	N/A	N/A
300	350	4.4	0.6	N/A	N/A
Average		11.5	2.0	10.3	5.1

the reference system stops working, whereas the proposed system still works providing consistent accuracy (11.8 pixels in average).

2.7.4 Computational Load Assessment

In order to assess the computational load, we did an online experiment during 10 minutes with two cameras and collected the time cost of the vision task and the pose computation separately, as listed in Table 2.9. The average frame rate is 10 frames/sec which fulfills the real-time requirement of most applications.

TABLE 2.9: Execution time of the system.

Function	Mean (s)	Min (s)	Max (s)
Marker detection	0.0963	0.0210	0.2230
Pose estimation	0.0001	0.0001	0.0003
Total time	0.0964	0.0211	0.2233

2.7.5 Discussions

A. Discussions about accuracy assessment

From the accuracy assessment in terms of position and orientation estimation comparing with the real measurements, we can find that the stereo vision system is accurate, providing several centimeters error in position estimation and few degrees in orientation estimation. The monocular vision system provides lower accuracy in position estimation as expected,

whereas the high accuracy in orientation estimation is maintained. However, the requirement of the dense of the camera deployment is largely reduced, as when the marker is in the view of one camera, the estimation is still acceptable.

The proposed stereo vision system is compared with the marker-based reference system in terms of the displacement of the projection of test points with known coordinates. Both systems have achieved an average error around 10 pixels. The proposed system has lower standard deviation. Besides, the proposed system has a longer working range.

When analyzing the error sources of these experiments, the following aspects rise as relevant ones:

- Camera lens distortion: All cameras are inevitable to have lens distortion. Even though the camera calibration could correct most of the distortion, still there are some errors, which in this case affect the performance of both the proposed and the reference system.
- Marker size: When the mobile device is far away from the webcams, the colored marker appears small in the image and it will be more difficult to detect the markers accurately. This is the main reason why the accuracy of our system degrades with the distance, as shown in Fig. 2.22.
- 3D reconstruction error amplification: The depth estimation tends to be inaccurate when the real point is far away from the cameras, because even small errors in the image plane translate to large errors in distance. Take the case of 250 cm-180° in the first experiment as example. If an one-pixel error is introduced in the estimation of the markers in the x axis, after the 3D reconstruction the estimated y value will change 12 mm.
- Sensor inaccuracy: The error in the rotation angle around X axis is totally determined by the sensor information and, from Table 2.6, it is visible that there is around 1 degree error in the average sensor information acquisition.
- Ground truth errors: The “ground truth” is measured manually and this will introduce some error due to human manipulation.

B. Discussions about the computational cost

In the computational load assessment, the average time for a loop running in the stereo vision system is around 0.1 s, including the image processing and the pose calculation. Without taking any measure for optimization, a frame rate of 10 frames/s fulfills the real-time requirements for most of the applications. However, in a multi-camera system, the computational cost increases proportionally with the number of cameras. The issue related with multi-camera management will be discussed in Chapter 4.

2.8 Conclusions

In this chapter, we have proposed a hybrid pose estimation approach based on a colored fiducial, mobile accelerometers, and multi-sensor data fusion techniques. Several different fusion approaches for the pose estimation have been proposed depending on the number of cameras that see the marker, including the monocular vision, stereo vision, multi-camera and occlusion handling solutions. The proposed system has been validated in terms of accuracy and computational cost. The experimental results show that the proposed system achieves an accuracy in the order of centimeters for the position estimation and few degrees for the orientation estimation, providing measurements in real-time. Compared with the marker-based reference system, this system eliminates the need of deploying markers in the environment, which may cause visual discomfort. In a wide area, a large number of markers have to be placed carefully and calibrated, which is tedious. With respect to the accuracy, both systems are accurate. However, the proposed system is more stable and provides longer working range. As previously underlined, the system is built on inexpensive cameras, low-cost accelerometers which are typically embedded in tablets, smartphones, or wearable devices, and a colored paper marker, being an attractive option in terms of cost. Specially, it does not need a dense camera deployment, as pose estimation based on a single camera (with reduced performance) is still possible. Additionally, the system is easy to deploy, requiring a minimum of setup and configuration to get running. Calibration processes (both for the inertial sensor and the cameras) can be quickly done. All in all, we can claim that the system can provide accurate and stable results. Besides, the system can be easily extended to a large-scale space.

One main limitation of the current version of the propose system is that it is able to track only one user. However, an extension of the system for multiple users is possible. In the future, we will study the problem of tracking multiple users. There are several potential solutions. For example, devices may be tracked in the 3D space and distinguished by their trajectories. The user's body information (e.g., color histogram) or different marker designs can also be applied to identify each user. A user-in-motion accuracy assessment of the proposed system will be also studied to validate the system even more thoroughly.

Chapter 3

Object Motion Tracking

3.1 Introduction

The aim of an object tracker is to estimate the state of a moving object over time despite detection errors (false negatives or false alarms), occlusions, and the presence of other objects. It is one of the most important components in a wide range of applications in computer vision and other research fields, such as aircraft navigation (Gustafsson et al., 2002), object tracking (Yilmaz, Javed, and Shah, 2006; Joshi and Thakore, 2012), motion tracking (Ligorio and Sabatini, 2013), and robotics (Chen, 2012). A typical tracking system is composed of two major elements: observations and filtering. A third element - behavior analysis based on the tracking results may be included in some systems. In the previous chapter, different approaches to object pose estimation at each time instant have been presented. In this chapter, we will deal with the problem of data filtering, taking into consideration the measurement noise, outliers, loss of measurements, etc. There are some considerable reasons to use filtering and prediction in the field of computer vision:

- In practice, the observations from sensors are always uncertain or noisy, leading to noisy estimations. Estimations will be smoothed by filtering.
- In case of loss of measurements due to occlusions, illumination changes, or out of camera view, the state of the object is predictable for a certain period. Therefore, the tracking maintains the motion's continuity.
- Due to environmental noise and illumination changes, there are unavoidable wrong detections, which cause abrupt changes in the end user's application. The filter is able to detect and remove outliers.
- The computational cost may be considerably reduced using prediction. Since the predictor tells the position of the object in the next frame, there is no need to process the entire image. Instead, only a certain part of the image will be processed.
- The system's robustness may be improved. As only a small area of the image will be processed, noise and other objects outside the area that may cause confusion will not

influence or even disturb the processing. Therefore, the wrong detection rate may be reduced.

- Tracking is a vital technique for data association in multi-user applications. The motion of each user is tracked separately. As long as the objects do not collide and the measurement rate is high enough, the certainty of user identification will be kept high.
- The predicted target state is a piece of essential information for another important research topic: optimal camera selection in multi-camera systems. This will be presented in the next chapter.

Target tracking can be complex due to loss of information, unknown behaviors of measurement devices or the process, outliers, complex object motion, and partial or full object occlusions. Numerous approaches for object tracking have been proposed. They differ from each other based on the way they approach the following questions (Yilmaz, Javed, and Shah, 2006): What are the relevant parameters (object status representation) for tracking? What is the measurement sensor? How should the motion, appearance, and shape of an object be modeled? For visual tracking, which image features should be used? The answers to these questions depend on the end use for which the tracking information is applied to. Many approaches are focused on visual feature tracking in 2D, such as template-based kernel tracking and silhouette tracking for non-rigid body. Good reviews of visual tracking can be found in Yilmaz, Javed, and Shah (2006) and Joshi and Thakore (2012).

In this chapter, we will look at the tracking problem from the point of view of a dynamic system and concentrate on the probabilistic techniques (Bar-Shalom, 1987), which have been widely used to model dynamic systems in the last decades. Systems are described by their states, state transition model, and measurement model. At each time instant t_k ($k \in 0, 1, 2, \dots$), the information characterizing the system is defined by a state vector $\mathbf{x}_k$. For example, for a moving target, the state vector may consist of its position and velocity. The state transition model describes (linear/nonlinear) dynamic properties of the system, i.e., how the states evolve in time, as $\mathbf{x}_k = f_k(\mathbf{x}_{k-1}, \mathbf{w}_k)$, where $\mathbf{w}_k$ is the process noise. The measurements $\mathbf{z}_k$ are related to the corresponding states through the measurement model as $\mathbf{z}_k = h_k(\mathbf{x}_k, \mathbf{v}_k)$, where $\mathbf{v}_k$ is the measurement noise. In order to take into account the modeling and measurement inaccuracies, the process noise and measurement noise represented by random variables are introduced into the dynamic system. The objective of tracking is to estimate the state $\mathbf{x}_k$ given all the measurements $\mathbf{z}_{1:k} = \mathbf{z}_1, \dots, \mathbf{z}_k$ up to that moment, whereas prediction is to estimate the state in the next time instant $\mathbf{x}_{k+1}$. Among the probabilistic techniques, the Kalman filter (KF), developed by Kalman in 1960 (Kalman, 1960), is recognized as an efficient and versatile estimator. It provides a new means of computing least-squares estimates using state-space methods and employing a recursive algorithm for incorporating new observations. Still, it is the most widely used estimation algorithm in modern systems theory and has been employed in solving a huge range of estimation problems in many different application fields. A review of previous work on filtering approaches is presented in Section 3.2.

In this context, two tracking algorithms based on the Kalman filter are proposed in this chapter to keep the continuity of the motion and predict the state for the next time instant. The first Kalman filter is designed to track pose (position and orientation), which is the direct information of interest. On the other hand, as we have explained in the previous chapter, the initial outputs of the pose estimation system, no matter how many cameras are involved, are the reconstructed 3D positions of the two reference points, together with pitch and roll rotation angles (estimated directly from accelerometers). Then, the position of the target and the yaw rotation angle are estimated based on the geometric constraints. Consequently, the other filter is applied to track the two reference points in 3D space, pitch and roll angles. For the sake of simplicity, we name the first filter *Pose Tracker* and the other filter *Marker Tracker*. The implementations of these two filters, including the state model and initialization are explained in Section 3.3.

It is worth noting that due to various uncontrollable system factors, the measurement errors are not exactly Gaussian and the motion modeling is not perfect due to the user's arbitrary movements. Therefore, the process error covariance matrices Q and the observation error covariance matrix R must be appropriately tuned so that the magnitudes of process and measurement noise are sufficiently large to compensate for motion modeling and measurement errors, while at the same time keeping the consistency of the estimation. It is important to model the measurement and process noise as close as possible to the actual values. Our approach to model the measurement error using the error propagation technique is presented in Section 3.4. Two typical approaches to model the process error applied to our filters are introduced in Section 3.5.

One important usage of a Kalman filter is to detect and remove outliers to avoid the abrupt changes in the output. In Section 3.6, a criterion for detection of outliers is explained. Later from the experimental results, we have found that due to the change of measurement sources (camera hand-offs), false outlier detection may occur. To this end, we have proposed a method to reduce the false positive caused by camera hand-offs, which is further validated by the experimental results.

In Section 3.7, firstly, the proposed measurement error model is validated by real tests. To evaluate the performance of the two proposed filters, a test in realistic situation is performed and experimental results are analyzed. The innovation sequence is used to check the consistency of the filter and detect outliers. Besides, the prediction is employed to obtain a region of interest in the next frame to be processed, instead of processing the entire image. In this way, the computational cost is reduced and the robustness may be improved.

In summary, the contributions of the proposed filters are a) they are specially designed for tracking object position and orientation in 3D space; b) an analytical measurement error model is derived and validated by real tests; c) the filters are able to detect outliers and reduce the false outlier detection rate by dealing with camera hand-offs; d) the filters are able to smooth data, improve the estimation accuracy and reduce the computational cost.

3.2 Related Work

Probabilistic state-space approaches are the most widely used approaches to model discrete-time dynamic models by using probability distributions to describe observation and process uncertainties. In this context, recursive filters based on Bayes' Theorem (recursive Bayesian filters) are theoretically optimal solutions for filtering. There are two steps in a recursive filter. The *prediction* step uses the dynamic equation and the already computed probability density function (pdf) of the state at the previous time instant to derive the prior pdf of the current state. Then the *update* step employs the likelihood function of the current measurement to compute the posterior pdf.

3.2.1 Kalman Filter Framework

The Kalman filter is named after Rudolph E. Kalman, who in 1960 published his famous paper describing a recursive solution to the discrete-data linear filtering problem ((Kalman, 1960)). It is over 50 years old but is still one of the most important and common data fusion algorithms in use today. The Kalman filter which is an optimal Bayesian filter to estimate the state of a linear system from a series of noisy measurements, where the state is assumed to be distributed by Gaussian noise. It is efficient and recursive, providing an analytical solution of the estimation problem. Besides, it has the advantage that it is light on memory (it does not need to keep any history other than the previous state), and it is very fast, making it well suited for real time problems and embedded systems.

The framework of the Kalman filter contains a state transition model and a measurement model. The state transition model assumes that the state of a system at a time instant k evolves from the prior state at time instant $k - 1$ according to

$$\mathbf{x}_k = F_k \mathbf{x}_{k-1} + B_k \mathbf{u}_k + \mathbf{w}_k, \quad (3.1)$$

where

- $\mathbf{x}_k$ is the state vector containing the terms of interest for the system (e.g., position, velocity, and heading angle) at time instant k ;
- $\mathbf{u}_k$ is the vector containing any control inputs (e.g., steering angle, throttle setting, and braking force);
- F_k is the state transition matrix which applies the effect of each system state parameter at time $k - 1$ on the system state at time k (e.g., the position and velocity at time $k - 1$ both affect the position at time k);
- B_k is the control input matrix which applies the effect of each control input parameter in the vector;

FIGURE 3.1: The ongoing discrete Kalman filter cycle. The *time update* projects the current state estimate ahead in time. The *measurement update* adjusts the projected estimate by an actual measurement at that time.

- $\mathbf{w}_k$ is the vector containing the process noise terms for each parameter in the state vector. The process noise is assumed to be drawn from a zero mean multivariate normal distribution with covariance given by the covariance matrix Q_k .

$$\mathbf{w}_k \sim \mathcal{N}(0, Q_k)$$

The measurement model relates the state vector to the measurement vector through a transformation matrix as follows:

$$\mathbf{z}_k = H_k \mathbf{x}_k + \mathbf{v}_k, \quad (3.2)$$

where

- $\mathbf{z}_k$ is the vector of measurements;
- H_k is the transformation matrix that maps the state vector parameters into the measurement vector;
- $\mathbf{v}_k$ is the vector containing the measurement noise terms for each observation in the measurement vector. Like the process noise, the measurement noise is assumed to be zero mean Gaussian white noise with covariance R_k .

$$\mathbf{v}_k \sim \mathcal{N}(0, R_k)$$

The Kalman filter is a recursive filter, which contains two steps in each loop: a prediction and a update step. The prediction step projects forward the current state and covariance to obtain a priori estimate of the state and covariance. When a new measurement arrives (including some amount of error), the state and the covariance are updated using a weighted average of the prediction and the new measurement, with more weight being given to estimates with higher certainty. The Kalman filter cycle is shown schematically in Fig. 3.1. In what follows, the notation $\hat{\mathbf{x}}_{n|m}$ represents the estimate of $\mathbf{x}$ at time n given observations up to and including a time $m \leq n$.

Time update (“prediction”)

1) Predicted (a priori) state estimate

$$\hat{\mathbf{x}}_{k|k-1} = F_k \hat{\mathbf{x}}_{k-1|k-1} + B_k \mathbf{u}_k. \quad (3.3)$$

2) Predicted (a priori) covariance estimate

$$P_{k|k-1} = F_k P_{k-1|k-1} F_k^T + Q_k. \quad (3.4)$$

Measurement update (“correction”)

1) Calculate innovation or measurement residual

$$\tilde{\mathbf{y}}_k = \mathbf{z}_k - H_k \hat{\mathbf{x}}_{k|k-1}. \quad (3.5)$$

2) Calculate the innovation covariance

$$S_k = H_k P_{k|k-1} H_k^T + R_k. \quad (3.6)$$

3) Calculate the optimal Kalman gain

$$K_k = P_{k|k-1} H_k^T (S_k)^{-1}. \quad (3.7)$$

4) Update the state estimate

$$\hat{\mathbf{x}}_{k|k} = \hat{\mathbf{x}}_{k|k-1} + K_k \tilde{\mathbf{y}}_k. \quad (3.8)$$

5) Update the covariance estimate

$$P_{k|k} = (I - K_k H_k) P_{k|k-1}. \quad (3.9)$$

3.2.2 Review of Pose Tracking Techniques

The Kalman filter is commonly used for vehicles guidance and navigation, particularly aircraft and spacecraft (Grewal, 2011). Furthermore, it is widely applied to time series analysis used in fields such as signal processing and economics (Wells, 2013). Besides, it has been extensively used in the vision community for object tracking (e.g., balls, faces, hands, and heads) and many computer vision applications (e.g., feature tracking, cluster tracking, and fusing data from different sources) (Bar-Shalom, Willett, and Tian, 2011; Li et al., 2010).

However, in many applications, systems have non-linear transition or measurement models, where the Kalman filter is not applicable. During the last decades, extensions and generalizations to the Kalman filter have also been developed to work on non-linear systems, such

as the extended Kalman filter (EKF)(Lefferts, Markley, and Shuster, 1982) and the unscented Kalman filter (UKF) (Julier and Uhlmann, 2004). The EKF adapts techniques from calculus, namely multivariate Taylor Series expansions, to linearize a model around a working point. The non-linear functions f_k and h_k are linearized by first order Taylor polynomials in the neighborhood of prior estimate at current time moment. UKF obtains often a better linear model through (non-random) sampling. In general, Kalman filtering techniques are computationally simple. However, all these techniques are limited to cases where the Gaussian-linear assumption is a suitable approximation. Instead, particle filters can deal with estimation problems that are non-linear, sophisticated, and analytically intractable. Moreover, they accept more general system models than non-linear Kalman filters, namely, particle filters do not require system noise to be Gaussian. On the downside, in order to make reliable estimates, it must use a reasonably large amount of particles, which makes it computationally heavier than the EKF. For this reason, Kalman filters are preferred in this thesis.

As we mentioned in the previous chapter, vision-based approaches are widely used for the pose estimation task. However, since visual measurements are usually affected by significant noise and disturbances due to temporal and spatial sampling, quantization of the image signal, lens distortion, etc., the accuracy of the position and orientation estimation may degrade. To this end, Kalman filtering techniques are usually applied to vision measurements for temporal filtering and reducing the effect of vision measurement noise. Some techniques require the computation of a tracking error defined directly in the space of the image features, i.e., tracking in 2D space. For example, the work by Pertile et al. (2014) tracks the position and the velocity of the detected marker in the image plane. Some approaches work in the Cartesian space. The state vector is typically composed of 12 elements - pose and its first derivatives, while the measurements are the detected features in the image plane, providing an implicit and recursive solution to obtain the optimal pose estimation. Since the mapping of 3D points into the image plane is nonlinear, an EKF is typically applied, in which the measurement equations are linearized around the current state estimates. For instance, in an early work by Wang and Wilson (1992), an image preprocessor system provides at least three image plane measurements of selected known features, whereas five non-coplanar object features are recommended for accurate estimation. These measurements are then injected into an EKF to determine the 3D relative parameters of a random moving object with respect to a camera for robot control. The authors of Lippiello, Siciliano, and Villani (2002) propose a method to track pose of moving objects from visual measurements of a fixed stereo camera system by combining detected features and a known computer-aided design (CAD) model. The predicted position and orientation of the object provided by an EKF are used to reconstruct a 2D model of the visual parts of the objects in the image plane to improve the feature points selection process and reduce the processing time. A set of EKFs are used in Yu, Wong, and Chang (2005), where a Kanade-Lucas-Tomasi (KLT) tracker is used to extract feature points. In the first step, an EKF is applied to track the object's pose and in the second step, a set of EKFs, one for each model point, are exploited for the refinement of the positions of the model features in the 3D space. Additionally, the prediction capability of the filter allows setting up a dynamic windowing technique of the image plane which may

sensibly reduce the time required for feature extraction.

Despite the advantages of EKF, it is necessary to know the statistics of the measurement and process noise in order to guarantee satisfactory performance. Poor noise estimate or inaccurate model of the dynamic system degrade the system performance, and may even cause divergence (Janabi-Sharifi and Marey, 2010). Several schemes have been proposed in the literature to deal with this problem. Adaptive extended Kalman filter approaches (AEKF) have been developed to estimate the covariance matrix dynamically and automatically (Lippiello, Siciliano, and Villani, 2007; Karasalo and Hu, 2011). Another issue with the application of an EKF is that it linearizes the state and measurement equations about the predicted state. However, the prediction is often inaccurate in practice. An iterated extended Kalman filter is applied to refine the estimate by re-linearizing the measurement equation around the most recent estimate iteratively to increase the accuracy, however, at the expense of higher computational cost (Shademan and Janabi-Sharifi, 2005).

Besides its popularity in reducing the effect caused by the visual measurement noise, the Kalman filter is also widely applied to sensor-based pose estimation approaches, thanks to its capability of statistically fusing data from different sensors. For instance, accelerometers, gyroscopes, and magnetometers are typically used to sense the accelerations, angular rates, and magnetic fields. Then, the Kalman filter is applied to fuse these measurements to get an optimal pose estimation (Yun et al., 2003; Sabatini, 2006). Another example is provided by Ligorio and Sabatini (2013), which fuses the measurements from a monocular vision system with inertial/magnetic measurements from an IMU rigidly connected to the camera using an EKF.

The Kalman filter works on the basis of a certain dynamical model of the system. However, in practice, the behavior of a target may not be characterized at all times by a single model, such as maneuvering targets. A well-known solution is to describe the behavior with multiple models (Li and Jilkov, 2005). One of the significant schemes is the Interacting Multiple Model (IMM) algorithm, initially proposed in Blom (1984), which consists of several parallel Kalman filters. Each individual filter models the target behavior with one model. Then, the IMM provides an optimal weighted sum of the output of all the filters. The IMM is the state-of-the-art approach for maneuvering target tracking, as it is able to rapidly adjust to target maneuvers and achieves an excellent compromise between complexity and performance (Mazor et al., 1998). The reader may refer to Blom (1984) and Pitre, Jilkov, and Li (2005) for a detailed description of the algorithm and a comparison between different types of IMM.

3.3 Tracking Strategy Overview

3.3.1 Pose Tracker: System and Observation Models

As we mentioned before, the Pose Tracker is employed to track directly the target's pose, which is the output of the object pose estimation component. The filter system architecture is depicted in Fig. 3.2. The system and observation models of the Pose Tracker are designed as follows.

FIGURE 3.2: Pose Tracker system architecture.

State vector: The state vector $\mathbf{x}$ is a 9×1 vector containing the position data with its first derivatives (velocity) and the rotation data in form of three Euler angles (roll, pitch, and yaw). It is assumed that the velocity is constant and the angles do not change between updates.

$$\mathbf{x} = (\mathbf{P}_w, \dot{\mathbf{P}}_w, \psi, \theta, \phi). \quad (3.10)$$

State transition model: The state transition model relates the state at a previous time $k - 1$ with the current state at time k , as described in Eq. 3.1. In this case, the control input Bu_k is discarded as there is no external input. The process model stays linear as

$$\mathbf{x}_k = F_k \mathbf{x}_{k-1} + \mathbf{w}_{k-1}, \quad (3.11)$$

where

$$F_k = \begin{pmatrix} I_{3 \times 3} & \Delta t I_{3 \times 3} & 0_{3 \times 3} \\ 0_{3 \times 3} & I_{3 \times 3} & 0_{3 \times 3} \\ 0_{3 \times 3} & 0_{3 \times 3} & I_{3 \times 3} \end{pmatrix}, \quad (3.12)$$

where $I_{m \times m}$ indicates an identity matrix of $m \times m$ and Δt is the time interval between updates.

Measurement model: The measurement model relates the current state to the measurement $\mathbf{z}_k$ with the matrix H . In this case, the measurement vector is a 6×1 vector, containing the

estimated 3D position and three rotation angles. The model is interpreted from Eq. 3.2 as

$$\mathbf{z}_k = \begin{pmatrix} \tilde{\mathbf{P}}_w \\ \tilde{\psi} \\ \tilde{\theta} \\ \tilde{\phi} \end{pmatrix} = \begin{pmatrix} I_{3 \times 3} & 0_{3 \times 3} & 0_{3 \times 3} \\ 0_{3 \times 3} & 0_{3 \times 3} & I_{3 \times 3} \end{pmatrix} \mathbf{x}_k + \mathbf{v}_k. \quad (3.13)$$

3.3.2 Marker Tracker: System and Observation Models

The Marker Tracker is designed to track the position of two reference points in 3D space as well as two rotation angles obtained from the accelerometers. Afterwards, the pose and the rotation angle yaw are calculated using the smoothed 3D positions of two reference points based on the geometric constraints. This process is depicted in Fig. 3.3. The system and observation models of the Marker Tracker are designed as follows.

FIGURE 3.3: Marker Tracker system architecture.

State vector: The state vector $\mathbf{x}$ is a 14×1 vector containing two reference points' positions in the world coordinate system $\mathbf{P}_w^{(1)}$ and $\mathbf{P}_w^{(2)}$ with their first derivatives (velocity), as well as the rotation angles pitch and roll. It is assumed that the velocities and the angles do not change between updates.

$$\mathbf{x} = (\mathbf{P}_w^{(1)}, \mathbf{P}_w^{(2)}, \dot{\mathbf{P}}_w^{(1)}, \dot{\mathbf{P}}_w^{(2)}, \psi, \theta). \quad (3.14)$$

State transition model: The state transition matrix is a 14×14 matrix.

$$\mathbf{x}_k = \begin{pmatrix} I_{3 \times 3} & \Delta t I_{3 \times 3} & 0_{3 \times 3} & 0_{3 \times 3} & 0_{3 \times 2} \\ 0_{3 \times 3} & I_{3 \times 3} & 0_{3 \times 3} & 0_{3 \times 3} & 0_{3 \times 2} \\ 0_{3 \times 3} & 0_{3 \times 3} & I_{3 \times 3} & \Delta t I_{3 \times 3} & 0_{3 \times 2} \\ 0_{3 \times 3} & 0_{3 \times 3} & 0_{3 \times 3} & I_{3 \times 3} & 0_{3 \times 2} \\ 0_{2 \times 3} & 0_{2 \times 3} & 0_{2 \times 3} & 0_{2 \times 3} & I_{2 \times 2} \end{pmatrix} \mathbf{x}_{k-1} + \mathbf{w}_{k-1}, \quad (3.15)$$

where Δt is the time interval between updates.

Measurement model: In this case, the measurement vector is a 8×1 vector, containing the estimated 3D positions of two reference points and two rotation angles.

$$\mathbf{z}_k = \begin{pmatrix} \tilde{\mathbf{P}}_w^{(1)} \\ \tilde{\mathbf{P}}_w^{(2)} \\ \tilde{\psi} \\ \tilde{\theta} \end{pmatrix} = \begin{pmatrix} I_{3 \times 3} & 0_{3 \times 3} & 0_{3 \times 3} & 0_{3 \times 3} & 0_{3 \times 2} \\ 0_{3 \times 3} & 0_{3 \times 3} & I_{3 \times 3} & 0_{3 \times 3} & 0_{3 \times 2} \\ 0_{2 \times 3} & 0_{2 \times 3} & 0_{2 \times 3} & 0_{2 \times 3} & I_{2 \times 2} \end{pmatrix} \mathbf{x}_k + \mathbf{v}_k. \quad (3.16)$$

3.3.3 Initialization

The purpose of initialization is to give some reasonable “guessed” values for the initial state $\hat{x}(0|0)$ and $P(0|0)$. Although in general the effect of initial conditions diminishes with time and it is not critical to long-term running, initialization is still of importance in data filtering for real systems (Durrant-Whyte, 2001), especially in case where the filter has to be re-initialized often. Therefore, the choice of a reasonably good initial estimate may improve the filter performance.

A simple method of obtaining a good initial estimate of state is to process the first few observations. In this work, the first measurement and the time stamp are saved. When the second measurement comes, the new measurements of position and Euler angles are used as the initial estimates, whereas the initial velocity vector is calculated based on the movement between two measurements as

$$\mathbf{v}_0 = \frac{\tilde{\mathbf{P}}_2 - \tilde{\mathbf{P}}_1}{\Delta t}, \quad (3.17)$$

where Δt is the time between these two measurements.

To initialize the state covariance, a simple method is to choose a constant α and set $P(0|0) = \alpha^2 Q(k)$ (the determination of $Q(k)$ will be presented in Section 3.5). Again however, the effect of this initial estimate diminishes with time (Durrant-Whyte, 2001). As a rule-of-thumb, the initial covariance matrix should be large enough to enclose any likely uncertainty in the initial estimate.

In some cases, the filter has to be re-initialized. For example, due to certain reason (occlusion or out of camera view), the measurement may be missed for several frames successively. When the loss of measurement is over a certain threshold, the filter is not reliable any more. A re-initialization should be conducted. The choice of the threshold depends on the measurement rate and applications. In our case, 10 is chosen (around 1 second).

3.4 Measurement Error Modeling

Although the Kalman filter algorithm itself appears to be general and simple, its successful implementation tends to be problem-specific, relying heavily on the system model. Besides,

the performance of the filter depends directly on the adequacy of the process and sensor models. Therefore, it is important to model precisely the errors.

In this section we will propose a procedure for modeling the measurement error and finally derive a complete analytical measurement error model, which is related to the error on the accelerometer and the reference point estimation. To develop the complete error model, we will start focusing on the input error modeling (accelerometers and reference point estimation). Taking advantage of these error models, a complete model of the estimation error based on the propagation of the previously described errors will be introduced.

The modeling of the whole pose error distribution would be extremely complex due to the presence of different error sources in the input error terms and to the different weighting of those terms by the uncertainty propagation model. Therefore, in the following, we will just focus on the modeling of the principal error sources and further derive the error covariance matrices.

3.4.1 Accelerometer Measurement Error Model

The measurement of accelerometers suffer from various error sources. As previously described in Section 2.4.2 in Chapter 2, the accelerometers are calibrated using the Six-Position Static Test method to estimate the bias and the scale factor in each axis. From the calibration results, we have also estimated the noise standard deviations of accelerometer measurements from those 500 samples, resulting values equal to $\sigma_{ax} = 4.5 \times 10^{-3} \text{ m/s}^2$, $\sigma_{ay} = 4.3 \times 10^{-3} \text{ m/s}^2$, and $\sigma_{az} = 4.6 \times 10^{-3} \text{ m/s}^2$.

3.4.2 Reference Point Estimation Error Model

The errors in the estimation of reference points' coordinates (u, v) are due to image acquisition and processing, and to the algorithm used to detect the reference points. Several factors affect the image acquisition and reference points extraction process, such as sensor spatial and intensity quantization, image noise or light flickering. More detailed explanations of those effects can be found in Di Leo, Liguori, and Paolillo (2011) and De Santo et al. (2004). In this section we will provide rough quantitative estimations of the estimation error due to the proposed image-processing algorithm.

In our early experiments it became evident that different sizes and orientations of the fiducial projection in the image resulted in different errors. From this basic idea we performed a simulation to assess this error. In this simulation we assumed that space quantification due to image resolution is the dominant error source, and discard all other error sources. The Bresenham's line algorithm is adopted for the quantization process. An example of a line segment expressed in the image after quantization is shown in Fig. 3.4a. The ideal reference point is the center of the border segment between the different color rectangular areas in the

fiducial. The extracted reference point using the process in Section 2.5.2, is almost equivalent to the centroid of an area which can be obtained dilating the border segment projection in the image with the kernel described in step 5 of Algorithm 1, as can be seen in Fig. 3.4b.

FIGURE 3.4: (A) A line segment expressed in the image after quantization. (B) An example of the line segment after dilation.

Depending on the border segment size and orientation, the error, defined as the difference between the estimated centroid and the ideal reference point projection, is different. In our real scenario experiments (with a camera resolution of 640×360 pixels) the shared edge covers between two pixels (fiducial far away from the camera, 4.5 m) and eight pixels (fiducial very close to the camera, 30 cm). Therefore we model the projection of the border segment as a segment with length (L) from 2 to 8 pixels and inclination (θ , with respect to horizontal) from 0 to 180 degrees (note the image symmetry allows us to avoid modeling the angles between -180 and 0 degrees). The error is then measured as the difference between the centroid of the dilated region and the center of the line segment. In order to estimate the average error and the standard deviation of the error, we define a very fine grid (10×10 samples) for the position of the ideal center within a pixel. Then, we calculate the average value and the standard deviation of the centroid estimation from results for the different center positions in the grid, for a given border length and a given angle with respect to the horizontal. Algorithm 2 is an implementation of the previously described process.

Results are shown in Fig. 3.5. Due to problem symmetry, the mean error in both u-axis and v-axis (in Fig. 3.5a and Fig. 3.5b) is zero. The standard deviation of u-axis and v-axis, expressed in pixels, are shown in Fig. 3.5c and Fig. 3.5d. It is quite a complex function of border projection length and inclination.

From this result we developed an analytical model interpolating the simulated deviation. The basic idea behind this analytical model was realizing that the simulated deviation had a distinct “periodical” pattern, due to the quantization effect. In a previous research for a somehow similar problem (Garcia Herrero et al., 2001), a model for one dimensional (let us

Algorithm 2 Reference point error modeling

```

1: Create a binary image of  $30 \times 30$  pixel with all the elements set to zero.
2: Run the loop, where  $L$  is the length of the line segment and  $\theta$  is the angle between the
   line segment and the horizontal u axis.
3: for  $L = 2; L \leq 8; L = L + 0.1$  do
4:   for  $\theta = \pi/180; \theta \leq \pi; \theta = \theta + \pi/180$  do
5:     for  $C_u = 14.55; C_u \leq 15.45; C_u = C_u + 0.1$  do
6:       for  $C_v = 14.55; C_v \leq 15.45; C_v = C_v + 0.1$  do
7:          $(u^{(1)}, v^{(1)}) = (C_u - 0.5L \cos \theta, C_v - 0.5L \sin \theta);$ 
8:          $(u^{(2)}, v^{(2)}) = (C_u + 0.5L \cos \theta, C_v + 0.5L \sin \theta);$ 
9:         quantized_image = quantize the line segment
10:        dilated_image = dilate the quantized_image
11:        centroid_blob = find the centroid of the dilated_image
12:        error = centroid_blob -  $[C_u, C_v]$ 
13:       end for
14:     end for
15:   end for
16: end for
17: Save the 2D error vector for later analysis in an error matrix
18: Rearrange all the errors we get from each loop.
19: Analyze the data statistic of the errors varying from segment length and inclination,
   averaging results of all  $C_u$  and  $C_v$  values.

```

call it u) centroid estimation was:

$$\sigma_u = \sqrt{(1 - 3 \langle L_u \rangle + 3 \langle L_u^2 \rangle)/12}, \quad (3.18)$$

where L_u is the ideal length in pixels of the 1D image, and operator $\langle \rangle$ denotes the fractional part of the real number L_u .

This periodical function is shown in Fig. 3.6, with maximum values equal to $1/\sqrt{12}$ pixels for integer L_u values, and minimum values equal to $1/(2\sqrt{12})$ pixels for values of L_u with fractional part equal to 0.5.

Any dilation on the basic blob has no effect on centroid estimation error. In our 2D problem the extension of our blob along u-axis can be calculated through the projection of the segment border length (called L previously) along this axis. Additionally, for most (big enough) inclinations, we are averaging the results in several rows, which results in a reduced error, and demands a more complex model. A pretty good interpolation of the standard deviation of u in terms of L and θ may be obtained, finally, as:

$$\sigma_u = \begin{cases} \sqrt{(1 - 3 \langle L|\cos \theta| \rangle + 3 \langle L|\cos \theta|^2 \rangle)/12} & \text{if } L|\cos \theta| < 1/2\sqrt{12}, \\ (1/2 + 1/2|\cos \theta|)\sqrt{(1 - 3 \langle L|\cos \theta| \rangle + 3 \langle L|\cos \theta|^2 \rangle)/12} & \text{otherwise.} \end{cases} \quad (3.19)$$

For v-axis, the same relation appears, substituting $\cos \theta$ by $\sin \theta$. These detailed models might be exploited if we had very good knowledge of L and θ . Once the pose of the fiducial

FIGURE 3.5: Reference point error model simulation result. (A) Error in x-axis. (B) Error in y-axis. (C) Standard deviation in x-axis. (D) Standard deviation in y-axis.

FIGURE 3.6: Standard deviation (in pixels) for 1D quantification.

in 3D is calculated, these values, related with projection geometry, might be calculated approximately. But for a rough model of the error we might use a simplified model, applicable in worst case, assuming a standard deviation of the form:

$$\sigma_u = (1/2 + 1/2|\cos \theta|)/\sqrt{12}, \quad \sigma_v = (1/2 + 1/2|\sin \theta|)/\sqrt{12}. \quad (3.20)$$

An even simpler model, in worst case, is assuming constant standard deviations:

$$\sigma_u = 1/\sqrt{12}, \quad \sigma_v = 1/\sqrt{12}. \quad (3.21)$$

The three models in Eq. 3.19, Eq. 3.20 and Eq. 3.21 might be used at different processing stages. The model in Eq. 3.19 needs too accurate control of the environment, and will just be difficult to apply in an operational environment. The model in Eq. 3.20 might be used to calculate covariances in real time, and the model in Eq. 3.21 is more adequate for worst-case analysis. As we are neglecting other sources of error, we will use Eq. 3.21 in the following.

An additional relevant conclusion was obtained from our simulation described in Algorithm 2: there is negligible cross covariance between u-axis and v-axis independently of L and θ . Finally, in the following sections we assume the error in the estimation of both projected reference points are independent. In fact it is not completely true, but it is very complex to model this correlation effect, and we will show in Section 3.7 that this lack of information is not too important for overall system accuracy assessment.

3.4.3 Determining Measurement Error Covariance Matrices

In the assumptions of Kalman filters that we mentioned previously, the measurement noise random processes $\mathbf{v}_k$ are uncorrelated, zero-mean white-noise processes with covariance matrix R_k . If the measurement vector contains m elements, the covariance matrix R_k is a $m \times m$ matrix. That is

$$E[\mathbf{v}_k \mathbf{v}_j^T] = \begin{cases} R_k & k = j, \\ 0 & \text{otherwise,} \end{cases} \quad (3.22)$$

where R_k is symmetric positive semi-definite matrix.

The Kalman filter does not change R , as it indicates the belief about the accuracy of the measurements. In practice, the measurement errors will certainly exist in any physical system. In case of a direct measurement from a sensor, it may be reported in the manual or be measured. In our case, the measurement vector contains estimated 3D position and rotation angles which are derived from vision sensors and accelerometers. As we have already modeled the errors in the sensors, the pose estimation error can be determined based on the error propagation rule.

In our proposed pose estimation algorithms, the inputs are the measurements from the mobile accelerometer $[g_{ax}, g_{ay}, g_{az}]^T$, and the detected fiducial reference points in camera coordinates. The output of the system is the measurement vector of Pose Track ($\mathbf{z}_k = [\tilde{\mathbf{P}}_w \ \tilde{\psi} \ \tilde{\theta} \ \tilde{\phi}]^T$) or Marker Tracker ($\mathbf{z}_k = [\tilde{\mathbf{P}}_w^{(1)} \ \tilde{\mathbf{P}}_w^{(2)} \ \tilde{\psi} \ \tilde{\theta}]^T$). Depending on the available cameras we have different inputs. We take the stereo vision system and the monocular vision system as example.

a) For the stereo vision system we have two pairs of reference points projections, coming from each of the cameras: $[u_L^{(1)}, v_L^{(1)}]$ and $[u_L^{(2)}, v_L^{(2)}]$ from the “left” camera, and $[u_R^{(1)}, v_R^{(1)}]$ and $[u_R^{(2)}, v_R^{(2)}]$ from the “right” camera. Summarizing, the algorithm input measurement

vector in this case results:

$$\mathbf{X}_{in} = [g_{ax}, g_{ay}, g_{az}, u_L^{(1)}, v_L^{(1)}, u_L^{(2)}, v_L^{(2)}, u_R^{(1)}, v_R^{(1)}, u_R^{(2)}, v_R^{(2)}]^T. \quad (3.23)$$

b) For the monocular vision system we have one pair of reference points projections: $[u^{(1)}, v^{(1)}]$ and $[u^{(2)}, v^{(2)}]$. So we may define the algorithm input measurement vector:

$$\mathbf{X}_{in} = [g_{ax}, g_{ay}, g_{az}, u^{(1)}, v^{(1)}, u^{(2)}, v^{(2)}]^T. \quad (3.24)$$

In all cases, we may summarize the estimation algorithm as a function relating the available measurement vector and the pose estimator:

$$\mathbf{z} = f(\mathbf{X}_{in}). \quad (3.25)$$

Of course, the function is different for each of the proposed pose estimation approaches. To analyze estimation error we first divide the error sources in two kinds. The first kind is related to the propagation of the errors in the available inputs to the estimator. The second kind is the systematic error (e.g., imperfect camera calibration), which lead to potential biases in the estimation.

To analyze the propagation of input errors, we relate the errors in the measurements and the errors in the estimation, as follows:

$$\mathbf{z} - \Delta\mathbf{z} = f(\mathbf{X}_{in} - \Delta\mathbf{X}_{in}), \quad (3.26)$$

where $\mathbf{z}$ is the measurement vector (with errors), and $\Delta\mathbf{z}$ is the measurement error, $\mathbf{X}_{in}$ is the measurement vector (with errors), $\Delta\mathbf{X}_{in}$ is the measurement error vector and therefore the term $(\mathbf{X}_{in} - \Delta\mathbf{X}_{in})$ would be an “ideal” measurement without errors.

From this equation we can make a first order Taylor approximation of the ideal pose, assuming it is in the vicinity of the estimated pose:

$$\mathbf{z} - \Delta\mathbf{z} = f(\mathbf{X}_{in}) - F_X \Delta\mathbf{X}_{in} + \dots, \quad (3.27)$$

where F_X is the Jacobian matrix of function $f(\cdot)$ evaluated at $\mathbf{X}_{in}$, which can be expressed as:

$$F_X = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_1}{\partial x_2} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \frac{\partial f_2}{\partial x_1} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_2}{\partial x_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \frac{\partial f_m}{\partial x_2} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}, \quad (3.28)$$

where m is the dimension of the measurement vector $\mathbf{z}$ and n is the dimensional of the input vector $\mathbf{X}_{in}$.

FIGURE 3.7: Uncertainty propagation model of the stereo vision system

FIGURE 3.8: Uncertainty propagation model of the monocular vision system.

It is worth mentioning that due to the heavy non-linearity of the pose estimation system (deriving pose from vision sensors and accelerometers), it is not practical to calculate the Jacobian matrix analytically. However, it can be approximated numerically by finite difference as:

$$\frac{\partial f_i}{\partial x_j} \approx \frac{f_i(x_1, x_2, \dots, x_j + h, \dots, x_m) - f_i(x_1, x_2, \dots, x_j, \dots, x_m)}{h}, \quad (3.29)$$

where h is a small number. There are some other approaches to approximate numerically the Jacobian matrix. In any case, the numerical approach is easy to implement and runs fast, which is a suitable solution in real systems.

Neglecting higher order terms we could therefore relate measurement and pose estimation error as follows:

$$\Delta \mathbf{z} \approx F_X \Delta \mathbf{X}_{in}. \quad (3.30)$$

So the estimated pose error is the result of the propagation of the input error (uncertainty) through a system specific uncertainty propagation function F_X . The resulting error model for the Pose Tracker is summarized in Fig. 3.7 for the stereo vision system and in Fig. 3.8 for the monocular vision system. It should be emphasized F_X is different for the monocular and stereo vision system (even its size is different, it is a 6×7 matrix for the monocular vision system and 6×11 for the stereo vision system).

From Eq. 3.30, using the expectation operation over errors, we may define the following relation between pose estimation bias ($\mathbf{b}_s$ in the following), and input measurement vector bias (denoted $\mathbf{b}_X$):

$$\mathbf{b}_z \approx F_X \mathbf{b}_X. \quad (3.31)$$

Also, we may define the following approximate relation between measurement error covariance (denoted R_X) and the resulting measurement error covariance (R_z):

$$R_z \approx F_X R_X F_X^T. \quad (3.32)$$

In order to complete the pose estimator error model we need to remember the input measurements first and second order statistics described in Section 3.4.1 and Section 3.4.2, and apply to them the uncertainty propagation process defined by Eq. 3.31 and Eq. 3.32.

For the stereo vision system the resulting biases would be zero if we are using calibrated accelerometers, as in our system. Otherwise the (unpredictable) bias would depend on unknown S_i and b_i terms described in Section 3.4.1, as follows:

$$\mathbf{b}_z = F_X [b_x + S_x g_{ax}, b_y + S_y g_{ay}, b_z + S_z g_{az}, 0, 0, 0, 0, 0, 0, 0]^T. \quad (3.33)$$

This equation might be used to find worst case biases for uncalibrated accelerometers.

Regarding the covariance matrix, applying Eq. 3.32 to the stereo vision system results in:

$$R_z = F_X \text{diag}(\sigma_{ax}^2, \sigma_{ay}^2, \sigma_{az}^2, \sigma_u^2, \sigma_v^2, \sigma_u^2, \sigma_v^2, \sigma_u^2, \sigma_v^2, \sigma_u^2, \sigma_v^2) F_X^T, \quad (3.34)$$

where $\text{diag}(\cdot)$ represents a diagonal matrix (in this case of size 11×11) whose diagonal elements are listed as parameters, and all standard deviations were introduced in Section 3.4.1 and Section 3.4.2. For σ_u and σ_v we will use in general the value predicted in Eq. 3.21.

An equivalent process might be performed for the stereo vision system, resulting again in an unbiased pose estimation for calibrated accelerometers, and in a biased pose estimation for uncalibrated accelerometers equivalent to:

$$\mathbf{b}_s = F_X [b_x + S_x g_{ax}, b_y + S_y g_{ay}, b_z + S_z g_{az}, 0, 0, 0, 0]^T. \quad (3.35)$$

Again, we can model the pose covariance for the monocular vision system as:

$$R_z = F_X \text{diag}(\sigma_{ax}^2, \sigma_{ay}^2, \sigma_{az}^2, \sigma_u^2, \sigma_v^2, \sigma_u^2, \sigma_v^2) F_X^T, \quad (3.36)$$

where all standard deviations were introduced in Section 3.4.1 and Section 3.4.2. For σ_u and σ_v we will use in general the value predicted in Eq. 3.21.

Until here, we have already completed the analytical measurement error model according to the error propagation rule, which is related to the error on the accelerometer and the reference point estimation. On one side, the error model is able to estimate the statistics of the measurement noise as close as possible to the actual values. On the other side, it is a useful tool for system stability assessment at any given position and any given orientation. We took the monocular and stereo vision system for the Pose Tracker as example. However,

the same process can be done to the Marker tracker and a multi-camera system. It has to be noted that we have considered a perfect camera model. However, due to the imperfection in the camera calibration process, they may introduce some amount of error, which is complicated to model. In Section 3.7 we will also use real error measurements to validate this model.

3.5 Process Error Modeling

A general model of motion is employed in our filter - constant velocity and angles. Under the assumptions of a Kalman filter, the process noise w_k is zero-mean and white with covariance matrix Q_k . We have applied two typical approaches to model the process error (Bar-Shalom, Li, and Kirubarajan, 2004) for our filter. The first one is obtained by discretizing the continuous-time state space representation driven by white noise. The other one is to define directly the state model in discrete time using a piecewise-constant white random sequence as process noise. For the sake of clarity and simplicity, in the derivation to obtain the process error covariance matrix, we will take the Pose Tracker as example, whereas the similar process can be done to the Marker Tracker, whose covariance matrix is provided at the end of each section.

3.5.1 Discretized Continuous-time Kinematic Models

The state vector of the Pose Track is composed of position, velocity and rotation angles as $\mathbf{x} = (\mathbf{P}_w, \dot{\mathbf{P}}_w, \psi, \theta, \phi)$. We assume that between the $k - 1$ and k time steps, the object is moving with a constant velocity and the rotation angles do not change. However, in practice, human motion will scarcely be of constant velocity and the orientation of the target can be changed, as it is held by a user. This can be modeled by continuous time zero-mean white noise.

Considering that the position and velocity in each axis are independent and are not correlated, we first start with one dimensional derivation, for example the x axis, and then expand it for the full state vector.

The continuous-time state equation is

$$\dot{x}(t) = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} x(t) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} w_x(t). \quad (3.37)$$

The discrete-time state transition model with time interval Δt is

$$\begin{bmatrix} X_w \\ \dot{X}_w \end{bmatrix}_k = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \begin{bmatrix} X_w \\ \dot{X}_w \end{bmatrix}_{k-1} + w_{xk}. \quad (3.38)$$

Therefore, the relation between the discrete-time process noise and the continuous-time one can be expressed as

$$w_{xk} = \int_{t_{k-1}}^{t_k} \begin{bmatrix} 1 & \Delta t - \tau \\ 0 & 1 \end{bmatrix} \begin{bmatrix} 0 \\ 1 \end{bmatrix} w_x(\tau) d\tau. \quad (3.39)$$

Please note that $w_x(t)$ is continuous-time zero-mean white noise, therefore,

$$E[w_x(t)] = 0, \quad (3.40)$$

$$E[w_x(t)w_x(\tau)] = \tilde{q}_x \delta(t - \tau), \quad (3.41)$$

where $\tilde{q}_x$ is the power spectral density of the continuous-time process noise $w_x(t)$. Later, we will see that this value needs to be tuned to fit the actual model.

The covariance matrix Q_{xk} of the discrete-time process noise w_{xk} can be obtained by using Eq. 3.41, assuming $\tilde{q}_x$ to be constant (A more detailed mathematical derivation can be found in Kohler et al. (1997)).

$$\begin{aligned} Q_{xk} &= E[w_{xk}w_{xk}^T] = \int_{t_{k-1}}^{t_k} \begin{bmatrix} \Delta t - \tau \\ 1 \end{bmatrix} \begin{bmatrix} \Delta t - \tau & 1 \end{bmatrix} \tilde{q}_x d\tau \\ &= \begin{bmatrix} \frac{1}{3}\Delta t^3 & \frac{1}{2}\Delta t^2 \\ \frac{1}{2}\Delta t^2 & \Delta t \end{bmatrix} \tilde{q}_x. \end{aligned} \quad (3.42)$$

The same process can be performed on the position and velocity in y and z axes. Now let us have a look at the rotation angles, starting with the angle ψ . The continuous-time state equation is

$$\dot{\psi}(t) = w_\psi(t). \quad (3.43)$$

The discrete-time process noise is related to the continuous-time one as:

$$w_{\psi k} = \int_{t_{k-1}}^{t_k} w_\psi(\tau) d\tau. \quad (3.44)$$

The covariance matrix $Q_{\psi k}$ of the discrete-time process noise $w_{\psi k}$ can be obtained as

$$Q_{\psi k} = E[w_{\psi k}w_{\psi k}^T] = \int_{t_{k-1}}^{t_k} \tilde{q}_\psi d\tau = \Delta t \tilde{q}_\psi, \quad (3.45)$$

where $\tilde{q}_\psi$ is the power spectral density of $w_\psi(t)$.

In summary, the complete process noise covariance matrix Q_k (9×9 matrix) for the Pose Tracker can be obtained as

$$Q_k = \begin{bmatrix} \frac{1}{3}\Delta t^3 \tilde{q}_x & 0 & 0 & \frac{1}{2}\Delta t^2 \tilde{q}_x & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{3}\Delta t^3 \tilde{q}_y & 0 & 0 & \frac{1}{2}\Delta t^2 \tilde{q}_y & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{3}\Delta t^3 \tilde{q}_z & 0 & 0 & \frac{1}{2}\Delta t^2 \tilde{q}_z & 0 & 0 & 0 \\ \frac{1}{2}\Delta t^2 \tilde{q}_x & 0 & 0 & \Delta t \tilde{q}_x & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{2}\Delta t^2 \tilde{q}_y & 0 & 0 & \Delta t \tilde{q}_y & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2}\Delta t^2 \tilde{q}_z & 0 & 0 & \Delta t \tilde{q}_z & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \Delta t \tilde{q}_\psi & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \Delta t \tilde{q}_\theta & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \Delta t \tilde{q}_\phi \end{bmatrix}, \quad (3.46)$$

where $\tilde{q}_i$, ($i = x, y, z, \psi, \theta, \phi$) is the power spectral density of the continuous-time process noise in the corresponding dimension.

The Q_k matrix for the Marker Tracker is a 14×14 matrix, which can be expressed as

$$Q_k = \begin{bmatrix} A & & & 0 \\ & A & & \\ & & \Delta t \tilde{q}_\psi & \\ 0 & & & \Delta t \tilde{q}_\theta \end{bmatrix}, \quad (3.47)$$

where A is a 6×6 covariance matrix for the positions and velocities

$$A = \begin{bmatrix} \frac{1}{3}\Delta t^3 \tilde{q}_x & 0 & 0 & \frac{1}{2}\Delta t^2 \tilde{q}_x & 0 & 0 \\ 0 & \frac{1}{3}\Delta t^3 \tilde{q}_y & 0 & 0 & \frac{1}{2}\Delta t^2 \tilde{q}_y & 0 \\ 0 & 0 & \frac{1}{3}\Delta t^3 \tilde{q}_z & 0 & 0 & \frac{1}{2}\Delta t^2 \tilde{q}_z \\ \frac{1}{2}\Delta t^2 \tilde{q}_x & 0 & 0 & \Delta t \tilde{q}_x & 0 & 0 \\ 0 & \frac{1}{2}\Delta t^2 \tilde{q}_y & 0 & 0 & \Delta t \tilde{q}_y & 0 \\ 0 & 0 & \frac{1}{2}\Delta t^2 \tilde{q}_z & 0 & 0 & \Delta t \tilde{q}_z \end{bmatrix}. \quad (3.48)$$

It is worth nothing that the values of the power spectral density $\tilde{q}_i$ of the process noise have to be tuned according to the engineering judgment to compensate the state transition modeling error. Some guideline for the choice of these values is explained in (Bar-Shalom, Li, and Kirubarajan, 2004).

3.5.2 Direct Discrete-time Kinematic Model

Another common kinematic model is directly defined in discrete time. The assumption is that between the $k-1$ and k time steps, the object undergoes a *constant acceleration* of $\mathbf{a}_k$ that is normally distributed, with zero mean and standard deviation $\sigma_{\mathbf{a}}$. At the same time, as we assume that the rotation angles do not change between the updates, the object undergoes a

constant angular velocity of ω_k (Please note that the denotation is different from the process noise w_k) that is normally distributed, with zero mean and standard deviation σ_ω . Therefore, the velocity is increased by $\mathbf{a}_k \Delta t$ between the two time steps, while the position is increased by $\mathbf{a}_k \Delta t^2 / 2$. The effect of the angular velocity to the angles is $\omega_k \Delta t$.

The state transition model for the **piecewise constant white acceleration model** can be deduced from Newton's laws of motion as

$$x_k = F_k x_{k-1} + G_k \begin{bmatrix} \mathbf{a}_k \\ \omega_k \end{bmatrix}, \quad (3.49)$$

where

$$G_k = \begin{bmatrix} \frac{\Delta t^2}{2} I_{3 \times 3} & 0_{3 \times 3} \\ \Delta t I_{3 \times 3} & 0_{3 \times 3} \\ 0_{3 \times 3} & \Delta t I_{3 \times 3} \end{bmatrix}. \quad (3.50)$$

so that, the process noise w_k can be expressed as

$$w_k = G_k \begin{bmatrix} \mathbf{a}_k \\ \omega_k \end{bmatrix}, \quad (3.51)$$

where $w_k \sim \mathcal{N}(0, Q_k)$.

The covariance of the process noise Q_k for the Pose Tracker is

$$Q_k = E[w(k)w(k)^T] = \begin{bmatrix} \frac{\Delta t^4}{4} \sigma_x^2 & 0 & 0 & \frac{\Delta t^3}{2} \sigma_x^2 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{\Delta t^4}{4} \sigma_y^2 & 0 & 0 & \frac{\Delta t^3}{2} \sigma_y^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{\Delta t^4}{4} \sigma_z^2 & 0 & 0 & \frac{\Delta t^3}{2} \sigma_z^2 & 0 & 0 & 0 \\ \frac{\Delta t^3}{2} \sigma_x^2 & 0 & 0 & \Delta t^2 \sigma_x^2 & 0 & 0 & 0 & 0 & 0 \\ 0 & \frac{\Delta t^3}{2} \sigma_y^2 & 0 & 0 & \Delta t^2 \sigma_y^2 & 0 & 0 & 0 & 0 \\ 0 & 0 & \frac{\Delta t^3}{2} \sigma_z^2 & 0 & 0 & \Delta t^2 \sigma_z^2 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & \Delta t^2 \sigma_\psi^2 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & \Delta t^2 \sigma_\theta^2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & \Delta t^2 \sigma_\phi^2 \end{bmatrix}, \quad (3.52)$$

where $\sigma_i (i = x, y, z, \psi, \theta, \phi)$ is the standard deviation of the assumed constant acceleration and angular velocity in each dimension.

Similarly, we can obtain the process noise covariance matrix for the Marker Tracker as

$$Q = \begin{bmatrix} A & & & 0 \\ & A & & \\ & & \Delta t^2 \sigma_\psi^2 & \\ 0 & & & \Delta t^2 \sigma_\phi^2 \end{bmatrix}, \quad (3.53)$$

where

$$A = \begin{bmatrix} \frac{\Delta t^4}{4}\sigma_x^2 & 0 & 0 & \frac{\Delta t^3}{2}\sigma_x^2 & 0 & 0 \\ 0 & \frac{\Delta t^4}{4}\sigma_y^2 & 0 & 0 & \frac{\Delta t^3}{2}\sigma_y^2 & 0 \\ 0 & 0 & \frac{\Delta t^4}{4}\sigma_z^2 & 0 & 0 & \frac{\Delta t^3}{2}\sigma_z^2 \\ \frac{\Delta t^3}{2}\sigma_x^2 & 0 & 0 & \Delta t^2\sigma_x^2 & 0 & 0 \\ 0 & \frac{\Delta t^3}{2}\sigma_y^2 & 0 & 0 & \Delta t^2\sigma_y^2 & 0 \\ 0 & 0 & \frac{\Delta t^3}{2}\sigma_z^2 & 0 & 0 & \Delta t^2\sigma_z^2 \end{bmatrix}. \quad (3.54)$$

For this model, σ_i ($i = x, y, z$) should be of the order of the maximum acceleration magnitude σ_{Mi} . σ_i ($i = \psi, \theta, \phi$) should be of the order of the maximum angular velocity σ_{Mi} . A practical range is $0.5a_{Mi} \leq \sigma_i \leq a_{Mi}$.

Remarks

Both the continuous-time white noise and the piecewise constant white noise models are, obviously, approximations. Nevertheless, with appropriate choices of their intensities, both can model the motion with an acceptable uncertainty (Bar-Shalom, Li, and Kirubarajan, 2004). The second model is more commonly used, as the process noise intensity in this model has physical meaning related to the motion (acceleration and angular velocity). Therefore, in this work, we use the direct discrete-time kinematic model.

Influence of the covariance matrices Q and R

The process noise covariance matrix Q indicates the process uncertainty due to the kinematic modeling error. When Q is large, the filter puts more weight on the measurements. The measurement noise covariance matrix R determines the uncertainty of the measurements. If R is high, the filter considers the measurements are not very accurate. If R is small, the filter will follow the measurements closely.

3.6 Outlier Detection

Due to wrong marker detections caused by environmental noise or illumination changes, before injecting the measurements into the Kalman filter update phase, a necessary process is to detect and reject outliers or very noisy observations. The Normalized Innovation Squared (NIS) test is employed, which is defined as:

$$q_k = \tilde{y}_k^T S_k^{-1} \tilde{y}_k, \quad (3.55)$$

where $\tilde{y}_k$ is the innovation or measurement residual and S_k is the innovation covariance.

It is worth noting that this quantity defines a squared Mahalanobis distance (Mahalanobis, 1936), which is a measure of the distance between a point P and a distribution D . If there are m independent, standard normal random variables, the sum of their squares is distributed

FIGURE 3.9: χ_m^2 pdf with six degrees of freedom together with the 99% confidence region.

according to the chi-squared distribution with m degrees of freedom. Each time a new measurement becomes available, q_k is computed. A confidence interval is defined to specify whether the measurement is reliable or not. Measurements that pass this test are injected into the filter to update the states as described before. Or else, they are discarded.

The confidence interval $[r_1, r_2]$ is defined by the hypothesis H_0 that the q sequence is χ_m^2 distributed with probability $1 - \alpha$. So $[r_1, r_2]$ has to be defined in the way that

$$P(q \in [r_1, r_2] | H_0) = 1 - \alpha. \quad (3.56)$$

Considering that the estimation results are accurate in our system, $\alpha = 0.01$ is chosen (i.e., define the 99% confidence region). From the chi-squared statistical tables, we can find that the confidence interval for pose tracker ($m = 6$) is

$$[r_1, r_2] = [0, x_6^2(0.99)] = [0, 16.812]. \quad (3.57)$$

The chi-square probability density function (pdf) with six degrees of freedom together with the 99% confidence region is shown in Fig. 3.9.

The confidence interval for marker tracker ($m = 8$) is

$$[r_1, r_2] = [0, x_8^2(0.99)] = [0, 20.090]. \quad (3.58)$$

FIGURE 3.10: Standard deviation of pixel measurements from different distances to the cameras.

3.7 Experiments

3.7.1 Measurement Error Model Validation

In this section we will validate experimentally the proposed measurement error model in Section 3.4. Again, the stereo vision and monocular vision approaches for pose estimation are taken as example. In the previous Section 2.7.3, an experiment was conducted to assess the accuracy of the pose estimation system in seven static positions with distinct distances to the cameras. In each position, 50 samples were captured. The data from this experiment is also used to validate the measurement error model. Firstly, the proposed reference point estimation error model is validated by comparing the standard deviations of reference point measurements with the inputs from the error model. In Fig. 3.10, the horizontal line corresponds to the standard deviation predicted by our input error models (Eq. 3.21 for reference point measurements). We can see that these errors are similar to those in the model, which tends to overestimate them. This is expected from the error model discussion and derivation in Section 3.4.2.

The system then calculates the pose vector based on the detected reference points and sampled accelerometer measurements. In each position, the standard deviation of pose is estimated from measurements, and the results are depicted in Fig. 3.11a, Fig. 3.11b for stereo vision system and Fig. 3.12a, Fig. 3.12b for monocular vision system (left camera). Meanwhile, corresponding standard deviations predicted by the error model are presented in Fig. 3.11c, Fig. 3.11d for the stereo vision system and Fig. 3.12c, Fig. 3.12d for the monocular vision system. We can find that the curve of experimental results is consistent with the standard deviations predicted by the error model, both for stereo vision system and monocular vision system. If we compare Fig. 3.11 and Fig. 3.12, we can see that the stereo vision system is much more accurate (expected errors in few mm instead of few cm level). From this experiment, we could conclude that the proposed measurement error model based on the error propagation rule is sufficient for estimating the covariance of the measurement noise in a Kalman filter. However, it has to be clarified that this model is not enough to estimate the bias due to some uncontrollable systematic factors, such as imperfect camera calibration.

FIGURE 3.11: Comparison of position and Euler angles between experimental results and predicted results in the stereo vision system. (A) Experimental position standard deviation. (B) Experimental rotation standard deviation. (C) Predicted position standard deviation. (D) Predicted rotation standard deviation.

3.7.2 Experimental Setup for Tracking Performance Analysis

In order to demonstrate the validity of our proposed filter in realistic situations, we have conducted experiments using four webcams deployed in the ceiling pointing inwards a room, as shown in Fig. 3.13. The resolutions of the cameras are set to 640×360 pixels. The intrinsic and extrinsic parameters of the cameras are calibrated beforehand. The marker is pasted on the upper border of a tablet device and the relative pose does not change during the experiment.

For the sake of analyzing the performance of the filter from different perspectives and easiness to compare with different parameter settings, an experiment is conducted offline. Cameras capture frames at 10 frames/s, which are saved in the server. Simultaneously, the measurements from the accelerometers are stored.

A user held the tablet and moved around in the scenario at his own speed. The whole sequences took around 18 seconds and 180 images were recorded. Afterwards, the positions of the reference points in each frame are estimated by Algorithm 1. The data from the visual sensors and the accelerometers are then fused to estimate the position and orientation of the device based on the pose estimation algorithms proposed in the previous chapter. Note that

FIGURE 3.12: Comparison of position and Euler angles between experimental results and predicted results in the monocular vision system. (A) Experimental position standard deviation. (B) Experimental rotation standard deviation. (C) Predicted position standard deviation. (D) Predicted rotation standard deviation.

FIGURE 3.13: 2D and 3D trajectories of the experiment. (A) 2D trajectory. (B) 3D trajectory.

in this test, the measurements from all the cameras are exploited to estimate the device's pose and the ground truth of the device's pose is not available.

The 2D and 3D estimated trajectories are shown in Fig 3.13a and Fig 3.13b. The number of cameras that detect the tablet/marker in each frame is depicted in Fig. 3.14. As we can see

FIGURE 3.14: The number of cameras that detect the marker on each frame.

FIGURE 3.15: The position and orientation estimation from the experiment. (A) Position estimation in x, y and z axes. (B) Rotation estimation in x, y and z axes.

from the figure, in most of the frames the marker is detected by two cameras and occasionally, the marker is seen by three cameras. The position and orientation estimation in each axis are shown in Fig. 3.15.

Note that the performance of the filter relies heavily on the choices of the parameters (process noise intensities) in the process noise covariance matrix Q . It has to be tuned appropriately to fit the model. In our test, the parameters in Q are set as follows: $\sigma_i = 1.4 \text{ m/s}^2$ ($i = x, y, z$), $\sigma_i = 4.0 \text{ deg/s}$ ($i = \psi, \theta$) and $\sigma_\phi = 6.3 \text{ deg/s}$ for both trackers.

3.7.3 Experimental Results

3.7.3.1 Checking Filter Consistency

As described in (Durrant-Whyte, 2001), typically, the only information that is available about a process is the observation sequence. Therefore, the practical way of studying a filter's behavior is to observe the *innovation* or the residual, which is the difference between the measurement and the value predicted by the filter, as defined in Eq.3.5. If a filter works correctly, the assumptions made in deriving the Kalman filter should be tenable. That is to say, the innovation sequence should be zero-mean and white with known covariance S_k . So we can verify that the filter is consistent by testing the innovation sequence for these properties. Furthermore, it can be used to tune the parameters to adjust the filter performance.

Test 1: Innovation magnitude bound test

This test is performed by checking whether the innovation sequence is zero mean and conform with the corresponding covariances S_k calculated by the filter. The consistency of the innovation sequence with its covariance is checked by verifying that the magnitude of the innovation is bounded by $\pm 3\sqrt{S_k}$ ($\pm 3\sigma$), which indicates approximately 99.7% of the values will lie within the bounding gate. The innovation sequence in each degree of freedom (3-dimensional position and 3-dimensional orientation) as well as the $\pm 3\sigma$ from this experiment is shown in Fig. 3.16 (Pose Tracker) and Fig. 3.17 (Marker Tracker).

As we can see from these two plots, the innovations of position and rotation in each dimension lie within their estimated 3σ bounds. Also, note that due to the initialization process that we did, the filter's initial transient period does not cause any inconsistency.

Test 2: Normalized Innovation Squared (NIS) test

In order to test the unbiasedness of the innovation sequence, we need to verify that the squared Mahalanobis distance q_k lies in the confidence interval $[r_1, r_2]$ defined in Section 3.6 with an acceptable probability. The NIS test results are shown in Fig. 3.18a and Fig. 3.18b for the Pose Tracker and the Marker Tracker respectively. As we can see from the plots, the hypothesis is indeed acceptable for both trackers.

Test 3: Innovation whiteness (autocorrelation) test

In discrete time, white noise is serially uncorrelated random variables with zero mean and finite variance. Therefore, to test that the innovation is white, we can exploit ergodicity to define the test statistic as a time-averaged auto-correlation (Durrant-Whyte, 2001)

$$r(\tau) = \frac{1}{N} \sum_{k=0}^{N-\tau-1} \tilde{y}_k^T \tilde{y}_{k+\tau}. \quad (3.59)$$

The autocorrelation is usually normalized by $r(0)$. The normalized auto-correlation of the

FIGURE 3.16: Measurement residuals for the x , y and z axes (solid red lines) and the corresponding 3σ bounds (dashed blue lines) from the Pose Tracker. (A) Measurement residuals for position estimates in the x , y and z axes. (B) Measurement residuals for rotation angles estimates around the x , y and z axes.

innovations for the test we have done are shown in Fig. 3.19 and Fig. 3.20. Note that it peaks at $\tau = 0$ and everywhere else is distributed randomly about zero. Again in our test the autocorrelation satisfies the hypothesis.

3.7.3.2 Dealing with Camera Hand-offs for Outlier Detection

As we mentioned in Section 3.6, the same metric NIS is employed for the detection of outliers. If q_k lies outside of the defined confidence interval, the measurement is discarded as outlier. It is easy to find several peaks in both Fig. 3.18a and Fig. 3.18b. However, in this test,

FIGURE 3.17: Measurement residuals for the x , y and z axes (solid red lines) and the corresponding 3σ bounds (dashed blue lines) from the Marker Tracker. (A) Measurement residuals for position estimates in the x , y and z axes. (B) Measurement residuals for rotation angles estimates around the x , y and z axes.

there is no wrong marker detection and also the input from the accelerometer is reliable. That is to say, all the measurements from this test are reliable and accurate. Therefore, the measurements that arrive at the moments when the peaks occur should not be discarded.

This behavior can be explained, because there were camera hand-offs at the instants where these peaks appear. In other words, the camera set involved in pose estimation has changed in these moments. As cameras are not perfectly calibrated and the error caused by the imperfect camera model is not included in the system error modeling, it is reasonable to have some biases when the camera hand-off happens. In order to reduce the false positive rate,

FIGURE 3.18: Normalized innovation squared and the corresponding 99% confidence region. (A) Pose Tracker. (B) Marker Tracker.

FIGURE 3.19: (A) Autocorrelation of the innovation from the Pose Tracker in x, y and z axes. (B) Autocorrelation of the innovation from the Pose Tracker in rotation angles about the x, y and z axes.

FIGURE 3.20: (A) Autocorrelation of the innovation from the Marker Tracker in x, y and z axes. (B) Autocorrelation of the innovation from the Marker Tracker in rotation angles about the x, y and z axes.

FIGURE 3.21: Normalized innovation squared and the corresponding 99% confidence region after dealing with camera hand-offs. (A) Pose Tracker. (B) Marker Tracker.

FIGURE 3.22: Normalized innovation squared and the corresponding 99% confidence region after introducing some outliers. (A) Pose Tracker. (B) Marker Tracker.

some measure has to be taken to deal with these moments. The solution that we propose is to re-initialize the filter when there is a camera hand-off. The posterior state estimate is reset as the new measurements of the position and the orientation, whereas the predicted velocity is kept the same. The posterior error covariance matrix is reset as the predicted covariance estimate, whereas the cross-variance of position, orientation and the velocity are set to zero.

The results after dealing with camera hand-offs are shown in Fig. 3.21a and Fig. 3.21b for the Pose Tracker and the Marker Tracker respectively. As is shown in the figures, the peaks in normalized innovation squared caused by the camera hand-offs are removed.

As a final test of our outlier detection method, we manually introduced some displacement errors in marker detection results in frame 50 and frame 100. The results, as shown in Fig. 3.22a and Fig. 3.22b, are as expected that the NIS quantities in frame 50 and frame 100 are much higher than the bounds. Thus, they are successfully identified as outliers.

FIGURE 3.23: The number of cameras that detect the marker on each frame when cam2 is deactivated.

3.7.3.3 Data Smoothing

We can find from Section 3.7.2 (Fig. 3.14) that the marker is detected by at least two cameras in each frame in the experiment. The original outputs from the pose estimation module are smooth, as the measurements from the stereo vision or multi-vision system are accurate. In order to analyze the performance of the proposed filters for noisy measurements (from the monocular vision system), one of the four cameras is deactivated. In this case, the marker is detected in all the frames in the full trajectory, however only by one camera most of the time. An example of the number of cameras that detect the marker when cam2 is deactivated is shown in Fig. 3.23. The original estimation results and filtered results from the three cameras as well as the results from four cameras are plotted in Fig. 3.24 for position and rotation in each axis. It is easy to find that the original measurements (green lines) in the range of frame 50 to frame 90 are noisy, as the marker is seen only by one camera from a large distance. The results after filtering (red lines) are smoother. The results from four cameras are used as the reference (blue lines).

In order to understand the behavior of the proposed filters in terms of data smoothing, noisy results are generated by deactivating one camera. Therefore, the average absolute errors in position and orientation estimation are measured by comparing the original results from three cameras and the filtered results to the reference. In total, four situations are considered by deactivating each single camera. The errors from both trackers are listed in Table 3.1 and Table 3.2. Besides, the improvement in accuracy is calculated by comparing the errors in the filtered results to the errors in the original results.

The experimental data from the tables show that the Pose Tracker is able to improve the estimation accuracy and smooth the data for both the position and rotation estimation. The

FIGURE 3.24: The position and orientation estimation from three cameras. (A) Position estimation in x, y and z axes. (B) Rotation estimation in x, y and z axes. Originals results are raw estimation from three cameras. Filtered results are filtered estimation from three cameras. The reference is the estimation from four cameras.

Marker Trackers can also improve the position estimation accuracy. However, in some situations, the rotation estimation may degrade. This can be explained as the rotation angle around the z-axis (yaw) is not tracked directly, instead calculated from the filtered positions of two reference points, as explained in Eq. 2.53. It is very sensitive to the relative positions of the two reference points, which are tracked by two independent filters. Small variations of behaviors of the two filters may affect the estimation of the yaw angle.

TABLE 3.1: Errors of the original results (estimation from three cameras) and the filtered results (filtered estimation from three cameras) from the Pose Tracker compared to the reference (estimation from four cameras).

cam0	cam1	cam2	cam3	Position error (mm)			Rotation error (deg)		
				Original results	Filtered results	Improvement	Original results	Filtered results	Improvement
×	✓	✓	✓	319.2	273.8	14.2%	3.54	3.31	6.4%
✓	×	✓	✓	263.0	101.2	61.6%	3.55	2.97	14.3%
✓	✓	×	✓	229.6	167.9	26.9%	2.23	2.17	2.7%
✓	✓	✓	×	103.6	72.3	30.2%	2.17	2.09	3.5%

TABLE 3.2: Errors of the original results (estimation from three cameras) and the filtered results (filtered estimation from three cameras) from the Marker Tracker compared to the reference (estimation from four cameras).

cam0	cam1	cam2	cam3	Position error (mm)			Rotation error (deg)		
				Original results	Filtered results	Improvement	Original results	Filtered results	Improvement
×	✓	✓	✓	319.2	279.0	12.6%	3.54	2.72	23.2%
✓	×	✓	✓	263.0	103.4	60.7%	3.55	3.49	1.5%
✓	✓	×	✓	229.6	171.9	25.1%	2.23	2.51	-2.3%
✓	✓	✓	×	103.6	72.1	30.4%	2.17	2.59	-19.6%

3.7.3.4 Generating the Region of Interest (ROI) Using Prediction

Providing the predicted information of where the target is expected to appear in 3D space in the next time instant, the location of the target in the image can also be “guessed” according to the known camera projection model. Therefore, it is not necessary to process the entire image. Instead, only a sub-image or ROI enclosing the marker is processed. Consequently, the computational cost is reduced. Besides, the system robustness may improve, as the environmental noise and other objects outside of the ROI that may cause confusion will not influence the processing.

The size and shape of the ROI may be determined in various ways. For example, a fixed size of 32×32 pixels is chosen in Kwon, Shin, and Paik (2006). In Kim and Kang (2014), the size of the window is set as three or four times of the size of the previously detected object. In this work, a rectangular ROI centered at the predicted position of the marker is adopted. The size of the ROI should be sufficiently large to make sure that the tracked marker points are inside the selected window. An adequate ROI can be approximated as the rectangle that encloses the marker when it is close to and faces the camera. In our case, a ROI of 150×150 pixels is sufficiently large. An example of the generated ROI from an entire image is shown in Fig. 3.25. If the projection of the predicted marker lays outside the image, an empty ROI will be generated, i.e., the image will not be processed in the next time instant.

We then compared the average execution time for approaches with and without using ROIs. The aforementioned sequence with four cameras was used for this test and the result is listed in Table 3.3. As is expected, by generating ROIs based on the prediction from the Kalman filter, the computational cost is largely reduced.

FIGURE 3.25: A generated Region of Interest using prediction (blue rectangle).

TABLE 3.3: Average execution time for approaches with and without using ROIs.

Approach	Process all (s)	Process ROIs (s) with the Pose Tracker	Process ROIs(s) with the Marker Tracker
Marker detection	0.2456	0.1520	0.1520
Pose estimation	0.0002	0.0002	0.0002
Tracking and generating ROIs	Null	0.0013	0.0020
Total	0.2458	0.1535	0.1542

3.7.4 Discussions

From the conducted experiments, we can find some behaviors of the proposed filters as described in the following.

- The experimental results validate that the proposed measurement error model derived by error propagation is a proper model.
- By analyzing the innovation sequence from the experiments, we can find that the Pose Tracker and the Marker Tracker are both consistent.
- Both filters are able to detect outliers and reduce the false detection due to the camera hand-offs by partially re-initializing the filters.
- Both trackers are able to smooth the estimation results. In case of noisy measurements caused by few number of cameras or low marker detection rate or outliers, the Pose Tracker is able to improve the accuracy both in the position and the orientation estimation. The marker tracker can also improve the position estimation. However, the orientation estimation, in particular, the yaw rotation angle may degrade, as it is calculated by two reference points after filtering. Small changes may cause a big error in the angle estimation.

- Both filters are able to reduce the computational cost by processing only a ROI generated by the prediction from the filters, instead of the entire image.

3.8 Conclusions

In this chapter, we have presented two KF-based algorithms to track the position and orientation of an object in 3D space and keep the continuity of the motion over time despite of outliers, measurement and process noise. One filter is designed to track pose and the other filter is designed to track the two reference points of the marker as well as two rotational angles. The filter design is explained in detail, including the dynamic system model, initialization, measurement and process error model determination. A complete system error model is analytically derived to obtain the measurement error covariance matrix, which is later validated by experimental results. An approach to the detection of outliers is presented. In particular, the false detection of outliers caused by the change of measurement sensors is dealt with. The tracking performance of the proposed filters is evaluated by real tests. It is validated that both filters are consistent by analyzing the innovation sequence. Besides, the filters are able to detect outliers, smooth data, improve estimation accuracy and provide prediction information for future usages (e.g., obtaining ROI and selecting an optimal camera set). Besides, the computational cost is largely reduced by using the ROIs generated by the filters. From the experimental results, it can be found that the performances of Pose Tracker and Marker Tracker are quite similar. However, the Pose Tracker is preferable, as the state vector has a lower dimension than the Marker Tracker and it can improve the estimation accuracy in both position and orientation.

One limitation of the current work is that the experiment is conducted by only one user in a simple situation. In the future, the performance of the filter will be evaluated thoroughly with more testing data and more complex motion models. Besides, for generating the ROIs, a fixed dimension of the rectangle is used, whereas the predicted covariance of the estimate may be useful to set up an adaptive window to generate the ROIs.

Chapter 4

Camera Selection in Multi-camera Systems

4.1 Introduction

Multiple cameras are demanded for a wide area surveillance, as each camera has its limited field of view. There are substantial advantages of using multiple cameras over using only a single camera. Firstly, the system can be extended to larger areas. Secondly, the accuracy may be increased by fusing data from cameras with overlapping fields of view. Furthermore, decision robustness may improve, as the multiple cameras provide views of the same object from different perspectives, which alleviates occlusion problems. In particular, recently with the development of image sensor technology and distributed processing, a new ubiquitous paradigm - Visual Sensor Network (VSN) has been developed and rapidly applied to various application fields. The reader is referred to Liu (2014) for a survey of applications of visual sensor networks. They consist of tiny visual sensor nodes (smart cameras), integrated with an embedded processor and a wireless transceiver, enabled to process the images locally and communicate with others.

Multi-camera systems bring considerable advantages. However, as the number of the cameras increases, the camera network becomes more complex and it introduces some challenges in camera management, including sophisticated data fusion, increased cost on computation, and communication. The challenges vary depending on the camera network architectures. For example, in a centralized camera network, image data from all the cameras are transmitted to a central server to be processed. Regardless of the communication media (wire or wireless), bandwidth is a severe challenge when a large volume of data have to be transmitted at the same time. Besides, the processing burden on the central server is aggravated. However, in a distributed camera network, these are not serious issues as the images can be processed locally in each camera node. Nevertheless, the memory and processing power may be limited when multiple tasks have to be performed. In addition, the communication may become complex due to the collaboration among the nodes, such as fusing data from different sources. On the other side, for battery-powered camera nodes, the limited life time is a critical problem.

To gather up, in a practical camera network, computational power, communication bandwidth, and energy are limited resources. Therefore, the advantages that a multi-camera network brings (e.g., the increased operational volume, tracking accuracy, and robustness) must be traded off against the increased cost on the resources. Researchers from different fields are making effort to either prolong the battery life or to upgrade the communication protocols, or to improve the image processing efficiency. In this chapter, we will concentrate on one of the active research fields - optimal camera selection, applied to target tracking. The principal basis of this research direction is that not all the cameras contribute equally to the tracking task at each time instant, depending on the relative position between the target and the cameras. To this end, a mechanism to select an optimal subset of cameras to accomplish real-time seamless tracking with minimized cost while maintaining a desired quality is on demand. Only selected cameras will be activated and perform the tracking task, whereas the non-selected cameras can be left idle or used for other tasks.

Various camera selection approaches have been proposed. Some of them define energy efficiency as quality metric, which is the primary challenge in energy-constrained camera networks (Dagher, Marcellin, and Neifeld, 2006; Yu and Sharma, 2010). Some work has dealt with the problem from the point of view of bandwidth management (Yang and Nahrstedt, 2005). A number of approaches focus on measuring the quality of the image data with user-defined criteria, such as the distance between the object and the center of the viewing frustum (Park, Bhat, and Kak, 2006) and the angle between the user's and the camera's viewing direction (Soro and Heinzelman, 2007). Probabilistic approaches are also investigated in some work to select cameras based on the predicted target state (Monari and Kroschel, 2010). A review of previous work is presented in Section 4.2. However, most of these aforementioned approaches are targeted for person tracking in 2D, which are not suitable for tracking mobile devices. The reasons are twofold. Firstly, the dimension of a mobile device is much smaller than a person, which indicates that the device is easily occluded by the human body or other objects. Secondly, the movement of a device is more flexible, having six degrees of freedom. Therefore, in this chapter, we will discuss the problem of camera selection for tracking a mobile device in a multi-camera system. To our knowledge, this is the first work that focuses on this problem, which can also be generalized to small object tracking.

Going into detail, let us assume a network which contains N cameras observing a scene. The objective of the camera selection mechanism is to determine a most suitable camera set S , who will be activated at the next time instant to perform the tracking task. The selection problem is divided into two sub-problems: a) who will "see" the target (more specifically, two reference points)? b) among all the cameras that will "see" the target, who will be finally selected?

The first problem is dealt with in Section 4.3. The predicted state of the target at the next time instant is obtained from the Kalman filter, as described in the previous chapter. Therefore, in a static and calibrated camera network, the positions of the reference points in each camera frame can be also calculated by the 3D-2D projection rule. It is easy to tell whether the projected points fall inside the image. However, this is not sufficient to decide whether

the camera will see the marker, as the marker is often and easily occluded by the user's body. It is sophisticated and time-consuming to take into account all the possible occlusions in the environment. Therefore, we will place main emphasis on the primary source of occlusion to a hand-held device - the user's body. In this context, we will provide several occlusion modeling approaches. Especially, the plane model and the rectangle model are explained in detail and then validated by experimental results. Cameras that pass these tests are considered as available cameras.

The second problem is how to select an optimal subset of cameras from the available cameras. Some researchers look at this problem in a game theoretic manner (Soto, Song, and Roy-Chowdhury, 2009; Li and Bhanu, 2011b). On one side, the collaboration between cameras can provide more accurate estimations than the results from a single camera, as we mentioned in Chapter 2. On the other hand, cameras need to compete in limited resources. Only cameras that contribute more to the tracking task will win/be chosen. This problem is described in Section 4.4. Various quality metrics can be defined, whereas in this work, the distance between the marker to the camera is adopted according to the observations from the experiments.

The performance of the proposed camera selection approach is evaluated experimentally in Section 4.5. In the test, the selection accuracy of the plane model, the rectangle model, and the approach without occlusion modeling are compared to the ground truth. Besides, the average execution time with the current real deployment in the laboratory is compared among these approaches.

4.2 Related Work

Camera selection in a multi-camera network is a relatively new research topic. A crucial component in a camera selection mechanism is to quantify the contributions of each camera based on certain criteria in order to achieve the optimal solution with reduced cost on resources. The choice of criteria is driven by applications. For example, in case of battery-powered camera nodes, the energy efficiency has to be primarily considered, whereas the selection is mainly determined by the quality of the captured image data for the cameras that do not have energy limitations. In general, most of the approaches try to solve the problem from the perspective of energy efficiency, bandwidth allocation, quality of image data, or statistics. Various algorithms have been proposed in literature (Rowaihy et al., 2007; Soro and Heinzelman, 2009; Li and Bhanu, 2011a).

Some approaches select cameras based on minimizing energy consumption, i.e., maximizing the lifetime of the sensors. For example, authors in Fang, Yang, and Xue (2012) formulate the problem as an online integer linear programming problem and apply an activation strategy algorithm for each time slot to obtain the maximum total utility, subject to resource constraints. In Dagher, Marcellin, and Neifeld (2006), inter- and intra-sensor correlation is

efficiently exploited to maximize the battery lifetime of the camera nodes. In Yu and Sharma (2010), the network lifetime is modeled as a stochastic random variable that depends on the camera coverage and therefore, the energy allocation problem is formulated as a max-min optimization problem.

In a network with centralized processing, a large volume of data has to be transmitted to the central server. This is a big challenge for the channel bandwidth. In Yang and Nahrstedt (2005), a bandwidth management framework is introduced to dynamically allocate the bandwidth to cameras. The bandwidth coordinator assigns as much as channel time as possible to cameras that detect high motions, whereas only enough channel time is assigned to the cameras that detect small motions to keep its minimum frame rate. Cameras that do not detect motions are put in the sleep mode. In this way, the utilization efficiency of the bandwidth resource for multiple video streaming is improved.

A number of approaches try to find the best camera set using the quality of the image data as metric, which is application-specific and may be defined in various ways. In Park, Bhat, and Kak (2006), the criterion used for defining favorable cameras is a simple model. That is, the closer the object is to the center of the viewing frustum, the more accurate the measurements are. Thus, they quantize the viewing frustum of each camera into smaller unit boxes. Based on the distance from the center of the sub-frustum where a 3D point locates to the center of the viewing frustum of the given camera, a sorted list of most favorable cameras for the given position is provided. Finally, a look-up table is generated to store information about the most favorable cameras for an given point in the viewing frustum of a given camera. This approach is straightforward. However, creating a look-up table entry for each possible target location for each possible number of targets lead to an impractically large look-up table (Tessens et al., 2014). Additionally, the size of the sub-frustum may also affect the camera selection performance, as a round operation is done on the given 3D point to obtain its corresponding look-up table entry. Similarly, another approach in Soro and Heinzelman (2007) applies the visual sensor network to a telepresence system. The user is able to move and see the monitored space from any desired viewpoint. Therefore, a straightforward method is to choose the camera with the minimum angles between the camera's axis and the user's view direction. Besides, the energy limitations of battery-powered camera nodes are taken into account by another metric and cameras with higher remaining energy are favored. This approach provides a trade-off between the quality of the reconstructed images and the network lifetime. In Wang, Wang, and Fang (2014), target detection is performed on each node. Then the most appropriate camera node is selected by a confidence measure, which is a function of the target blob size and the posterior covariance estimate of the target blob in the frame. However, only one camera is assigned to estimate the target's final position, which may cause inaccuracy in the estimation due to measurement noises. The principal view, i.e., the view that contributes most to the desired observation of a scene, is determined by the number and the quality of detected faces in Tessens et al. (2008). A local quality-of-view metric, defined as the ellipse fit root mean square root, is proposed in Shen and Hornsey (2011), which is however only suitable for tracking spherical targets. Similarly, an approach

by minimizing the mean squared error of the object position estimation is proposed in Ercan, El Gamal, and Guibas (2006).

These aforementioned approaches based on the quality of the image data have some similarity: firstly, the image processing task is performed on all the images, either in the central server or locally; then the quality is measured upon the detection results. The downside of this framework is that the computational cost is high, as all the images need to be processed. To this end, some approaches have employed the probabilistic approach to predict where the target would be in the next frame and only turn on the cameras that would have the target in view. For example, in Monari and Kroschel (2010), the predicted target position is combined with the geometric model of the camera sensing ranges and the building map to select cameras. In Pahalawatta, Pappas, and Katsaggelos (2004), authors firstly use low-power sensors (e.g., passive infra-red) to detect the proximity of the target and trigger its neighboring cameras. The information utility is measured as the trace of the posterior covariance matrix of the estimate from an unscented Kalman filter, based on the fact that a larger covariance in the posterior distribution implies a greater uncertainty in the estimate given by the tracker. Then, the camera selection problem is formulated as maximizing the sum of the information utility provided by the active sensors, subject to a constraint on the average energy that can be used in the network. A dynamic node collaboration scheme is proposed in Liu, Zhang, and Ma (2009). Firstly, the camera node with the highest probability to detect the target is elected as the cluster head. Then, an optimization-based algorithm is used to select the cluster members to estimate the target's location cooperatively. Adopting the aforementioned dynamic node collaboration scheme, authors in Wang and Wang (2013) try to balance the trade-off between the information utility (tracking quality) and the remaining energy of nodes (network lifetime). All the nodes are used for detection, whereas only selected nodes participate in the tracking task. A general drawback of the probabilistic approaches is that the selection performance heavily depends on the tracking quality.

Game theory is also applied to the camera selection problem by some researchers (Soto, Song, and Roy-Chowdhury, 2009; Li and Bhanu, 2011b), considering the fact that the cameras in the camera network need collaboration for the tracking task, whereas they are competitors, subject to the resource constraint. Soto, Song, and Roy-Chowdhury (2009) deals with the problem of tracking multiple targets in a distributed network of self-configuring pan-tilt-zoom cameras. A camera control framework based on game theoretical ideas is introduced, which allows for viewing some targets at a high resolution while keeping the entire area under surveillance at an acceptable resolution. A camera selection approach which is independent of the camera topology is presented in Li and Bhanu (2011b). The bargaining mechanism for collaborations and resolving conflicts among the available cameras is based on the defined utilities.

In summary, diverse approaches have been proposed in literature with different focuses. Most of them are targeted for person tracking or event detection in environmental monitoring. Tracking a mobile device is more complicated than tracking a person, as it is small, therefore easily occluded by the human body or other objects. Besides, it has six degrees of

TABLE 4.1: Variables used in camera selection.

Variables	Meanings
Π_b	the user's body plane
$\mathbf{P}_d = [x_d, y_d, z_d]^T$	the 3D position of the device in the world coordinate system
$\mathbf{P}_c = [x_c, y_c, z_c]^T$	the 3D position of the camera in the world coordinate system
$\mathbf{P}_b = [x_b, y_b, 0]^T$	the body center on the ground plane in the world coordinate system
$\hat{\mathbf{P}}_1 = [\hat{x}_1, \hat{y}_1, \hat{z}_1]^T$	the predicted 3D position of the reference point 1 in the world coordinate system
$\hat{\mathbf{P}}_2 = [\hat{x}_2, \hat{y}_2, \hat{z}_2]^T$	the predicted 3D position of the reference point 2 in the world coordinate system
$\mathbf{n}_b = [n_x, n_y, 0]^T$	the normal vector of the user's body plane in the world coordinate system
$\mathbf{x}_d$	the unit vector aligned with the device's X-axis in the world coordinate system
D_0	the distance between the device and the user

freedom in the movement. Our emphasis is on proposing a camera selection mechanism for tracking a mobile device seamlessly with minimized cost in a multi-camera system. To our knowledge, this is the first work that focuses on this problem. The proposed mechanism can also be extended to other small object tracking.

4.3 Determining Available Cameras

Let us assume a multi-camera system consisting of N cameras. The cameras are mounted on the ceiling or the walls of a certain space (for example, a room or an art gallery), pointing toward the space's interior. All the cameras are static and calibrated. We assume that a user is walking through the space holding a mobile device with any arbitrary position and orientation. The network is aimed to provide real-time seamless tracking of the mobile device with minimum cost on communication and computation. The following description will explain the first step in the proposed camera selection framework to determine the available camera set, referred to the cameras that will "see" the marker. The first sub-step is to check whether the projections of the predicted reference points $\hat{\mathbf{P}}_1$ and $\hat{\mathbf{P}}_2$ will fall inside the image. For the sake of clarity, we name this test as *point-in-view test*. As we mentioned before, the mobile device is easily occluded by the user's body. To this end, we have proposed several approaches to model the human body based on the geometric constraints and check whether the body model will occlude the marker. This test is called *occlusion test*. Cameras that pass both the point-in-view test and the occlusion test are labeled as available cameras. Variables that will be used in the rest of the chapter are listed in Table. 4.1 and the geometric relation is represented in Fig. 4.1.

4.3.1 Point-in-view Test

Each camera has a directional field of view, which is a three-dimensional viewing frustum as shown in Fig. 4.2a. The near and far planes, which are perpendicular to the principal axis,

FIGURE 4.1: The geometric relation of the user, the device, and the camera.

FIGURE 4.2: The pinhole model of a camera. (A) The viewing frustum of a camera enclosed by the near plane and far plane. (B) The projection of a 3D point into the image plane.

set boundaries in the viewing volume, as all physical cameras have a limited depth of field. In a pinhole camera model, points in front of the camera are projected into the image plane along the ray from the point to the optical center, as shown in Fig. 4.2b.

Let us assume that the image resolution is set as $W \times H$ pixels and the camera transformation matrix M is obtained from the calibration. The tracking results from Chapter 3 provide us the predicted 3D positions of two reference points, denoted by $\hat{P}_1$ and $\hat{P}_2$ in this chapter. The point-in-view test is done in a simple way.

Firstly, the projections of the two 3D reference points in the image plane are calculated as

$$\lambda_i \begin{bmatrix} u_i \\ v_i \\ 1 \end{bmatrix} = M \begin{bmatrix} \hat{\mathbf{P}}_i \\ 1 \end{bmatrix}, \lambda = 1, 2, \quad (4.1)$$

where λ_i is a scale factor, (u_i, v_i) are the coordinates of the projection in the image plane (the origin lies on the left-up corner of the image).

The requirement to pass the point-in-view test is that both projections lie in the interior of the image. It can be achieved by simply comparing the coordinates of the projections to the image resolution as

$$0 < u_1 < W \quad \& \quad 0 < v_1 < H \quad \& \quad 0 < u_2 < W \quad \& \quad 0 < v_2 < H. \quad (4.2)$$

This process is repeated for each camera. The cameras that pass this test are considered as the available camera candidates. If no camera passes the test, the selection process is terminated. Otherwise, the process will continue, as the six-DoF pose can be retrieved from at least one camera.

4.3.2 Occlusion Test

It is not sufficient to make the decision of whether the camera will see the marker by the point-in-view test, as the marker is often and easily occluded by the user's body. It is sophisticated and time-consuming to model the human body meticulously. Here, we introduce several modeling approaches for the human body with different levels of complexity: a plane model, a rectangle model, and others. Regardless of the model that is used, two basic parameters are necessary to model the user's body: the position and the walking direction of the user. One possible solution to obtain this information is to add a person tracking module parallel to the marker tracking. However, it will introduce more complexity to the system and the computational cost will be heavier. Our approach is to estimate these two parameters based on the geometric assumptions of the relative pose of the user to the device, as shown in Fig. 4.1. The assumptions are as follows:

- a) the user is always holding the device in front of him;
- b) the distance between the device and the user is kept constant as D_0 .

Please be aware that these assumptions are not always tenable, however as observed from the experiments, the camera selection task is not very sensitive to these conditions. Besides, the camera selection task in this work is a tool to save computation and communication cost to provide real-time tracking with acceptable accuracy. That is to say, the cost for the camera selection procedure should be negligible compared to the cost of processing all the images.

A. Modeling the body as a plane

A simplified method to model the human body is to assume the body as an infinite plane. The plane separates the 3D space into two halves. Only when the device and the camera lie on the same side of the plane, the marker will not be occluded by the plane. This requirement can be interpreted geometrically as an intersection problem of the body plane and the line passing $\mathbf{P}_c$ and $\hat{\mathbf{P}}_1$ (or $\hat{\mathbf{P}}_2$), where $\mathbf{P}_c$ is the location of the camera projection center in the world coordinate system.

1) Body plane expression

In order to express the *body plane* Π_b , the normal vector $\mathbf{n}_b$ and a point in the plane are required. The first assumption may be interpreted as: the body plane is parallel to the vertical plane where the device's X-axis (as shown in Fig. 2.6) lies. The unit vector of the device's X-axis ($\mathbf{x}_d$) can be obtained from the rotation matrix R_{wa} , which transforms a vector expressed in $\{w\}$ to $\{a\}$ as shown in Eq. 2.1. Therefore, $\mathbf{x}_d$ can be obtained by combining Eq. 2.6 as

$$\mathbf{x}_d = R_{wa}^{-1} \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} \cos \theta \cos \phi \\ \cos \theta \sin \phi \\ -\sin \theta \end{bmatrix}. \quad (4.3)$$

From this equation, it can be found that the vertical plane where $\mathbf{x}_d$ lies (determined by the x and y elements of $\mathbf{x}_d$) is only affected by the rotation around the z-axis (the rotation angle ϕ). Based on this fact, the expression of the normal vector $\mathbf{n}_b$ can be obtained in a simple way. When the orientation of the world coordinate system is identical to the device coordinate system, which means there is no rotation, $\mathbf{n}_b$ is aligned with the y-axis as

$$\mathbf{n}_b = [0, 1, 0]^T. \quad (4.4)$$

When the device is rotated around z-axis by ϕ (ϕ is positive in counterclockwise), it takes the same effect on $\mathbf{n}_b$, as shown in Fig. 4.3. Therefore, the expression of $\mathbf{n}_b$ after rotation can be calculated as

$$\mathbf{n}_b = \begin{bmatrix} n_x \\ n_y \\ 0 \end{bmatrix} = \begin{bmatrix} -\sin \phi \\ \cos \phi \\ 0 \end{bmatrix}. \quad (4.5)$$

To express the body plane, a point on the plane is necessary, apart from the already calculated normal vector. Now, let us calculate the body center on the ground $\mathbf{P}_b$. The horizontal distance between $\mathbf{P}_b$ and the device position $\mathbf{P}_d$ is assumed as a known value D_0 . This can be expressed as

$$(\mathbf{P}_d - \mathbf{P}_b) \cdot \begin{bmatrix} 1 \\ 1 \\ 0 \end{bmatrix} = D_0 \mathbf{n}_b. \quad (4.6)$$

FIGURE 4.3: Rotate the normal vector $\mathbf{n}_b$ by ϕ .

Combining Eq. 4.5 and Eq. 4.6, the position of the body center on the ground plane ($z = 0$) can be obtained

$$\mathbf{P}_b = \begin{bmatrix} x_d + D_0 \sin \phi \\ y_d - D_0 \cos \phi \\ 0 \end{bmatrix}. \quad (4.7)$$

Therefore, the body plane Π_b can be determined by the normal vector $\mathbf{n}_b$ and $\mathbf{P}_b$ as

$$\mathbf{n}_b \cdot (\mathbf{P} - \mathbf{P}_b) = 0, \quad (4.8)$$

where $\mathbf{P}$ is any given point in the plane.

2) Line-plane intersection

In 3D space, a line l is either parallel to a plane or intersects it in a single point or is contained in the plane, as depicted in Fig. 4.4. We will take the point $\hat{\mathbf{P}}_1$ as example and the same process can be repeated on the other point. Let l be the line determined by the camera projection center $\mathbf{P}_c$ and the predicted reference point $\hat{\mathbf{P}}_1$. Then, l can be expressed by a parametric equation as

$$\mathbf{P}(s) = \mathbf{P}_c + s(\hat{\mathbf{P}}_1 - \mathbf{P}_c), \quad (4.9)$$

where the parameter s is a real number.

We first check the case A and C in Fig. 4.4 by testing whether the line direction vector is perpendicular to the plane normal vector as

$$(\hat{\mathbf{P}}_1 - \mathbf{P}_c) \cdot \mathbf{n}_b = \begin{cases} 0 & \rightarrow \text{parallel or in the plane} \\ \text{others} & \rightarrow \text{non-parallel} \end{cases} \quad (4.10)$$

FIGURE 4.4: The spatial relation between a line l and a plane Π . (A) No intersection. (B) One intersection point. (C) Line contained in the plane.

In practice, the body plane is a vertical plane and the user holds the device in front of him. It is impossible that the line lies totally in the body plane. If the line and plane are parallel, there is no occlusion. Otherwise, l and the body plane intersect in a unique point P_I . The following condition is met as P_I and P_b are in the body plane

$$\mathbf{n}_b \cdot (\mathbf{P}_I - \mathbf{P}_b) = 0. \quad (4.11)$$

Combining Eq. 4.9 and Eq. 4.11, s can be solved as:

$$s = \frac{\mathbf{n}_b \cdot (\mathbf{P}_b - \mathbf{P}_c)}{\mathbf{n}_b \cdot (\hat{\mathbf{P}}_1 - \mathbf{P}_c)}. \quad (4.12)$$

If $0 \leq s \leq 1$, $\mathbf{P}_I$ is a point on the finite segment $\mathbf{P}_c \hat{\mathbf{P}}_1$, which indicates the body plane lies between the camera projection center and the reference point. Therefore, we can conclude that the point is occluded by the body plane.

Else if $s < 0$, $\mathbf{P}_I$ is outside the segment on the side of $\mathbf{P}_c$. Therefore, the marker point is on the back of the camera. There is no view of the marker point.

Else if $s > 1$, $\mathbf{P}_I$ is outside the segment on the side of $\hat{\mathbf{P}}_1$. There is no occlusion.

The relationship is shown in Fig. 4.5. At the end of this step, we can summarize the results as follows

$$\begin{cases} s < 0 & \rightarrow \text{no view} \\ s > 1 & \rightarrow \text{no occlusion} \\ 0 \leq s \leq 1 & \rightarrow \text{occlusion} \end{cases} \quad (4.13)$$

B. Modeling the body as a rectangle

In a special case, when the modeled body plane is in the middle of the camera and the marker, the camera may still see the marker. An example is shown in Fig. 4.6, where the camera can see the marker through the head. To this end, we propose another more accurate

FIGURE 4.5: The intersection of a ray and a plane.

FIGURE 4.6: A special case that the plane model does not work.

approach to model the user's body. A 2D rectangle with a dimension of $L \times H$ is employed to enclose the body, as shown in Fig. 4.7a. An example of the rectangle model in real test is depicted in Fig. 4.7b. Instead of making the final decision that there is occlusion when the body plane lies between the test point and the camera, a further step is taken to check whether the intersection lies inside the rectangle.

1) Calculate rectangle corners

Firstly, the 3D positions of the four corners of the rectangle are denoted by A , B , C and D with the coordinates as follows:

$$A = \begin{pmatrix} x_A \\ y_A \\ z_A \end{pmatrix}, B = \begin{pmatrix} x_B \\ y_B \\ z_B \end{pmatrix}, C = \begin{pmatrix} x_C \\ y_C \\ z_C \end{pmatrix}, D = \begin{pmatrix} x_D \\ y_D \\ z_D \end{pmatrix}. \quad (4.14)$$

Based on the geometrical constraint that A and B are on the ground and the knowledge of the body length L and the body height H , the coordinates of these four points can be

FIGURE 4.7: The rectangle and cuboid occlusion models. (A) The rectangle model encloses the body with a dimension of $L \times H$. (B) The rectangle model used for modeling the user's body. (C) The cuboid model used for modeling the user's body.

expressed as:

$$A = \begin{pmatrix} x_A \\ y_A \\ 0 \end{pmatrix}, B = \begin{pmatrix} x_B \\ y_B \\ 0 \end{pmatrix}, C = \begin{pmatrix} x_B \\ y_B \\ H \end{pmatrix}, D = \begin{pmatrix} x_A \\ y_A \\ H \end{pmatrix}. \quad (4.15)$$

As mentioned earlier, the position of the body center on the ground is calculated as $\mathbf{P}_b = [x_b, y_b, 0]^T$. Therefore the vector from A to $\mathbf{P}_b$ is

$$\mathbf{AP}_b = [x_b - x_A, y_b - y_A, 0]^T. \quad (4.16)$$

$\mathbf{AP}_b$ is perpendicular to the normal vector $\mathbf{n}_b$, leading to

$$\mathbf{AP}_b \cdot \mathbf{n}_b = 0. \quad (4.17)$$

Besides, the magnitude of the vector $\mathbf{AP}_b$ is $L/2$

$$\|\mathbf{AP}_b\| = L/2. \quad (4.18)$$

Therefore, the values of x_A and x_B can be solved from the system of equations composed of Eq. 4.17 and Eq. 4.18. However, the Eq. 4.18 is a quadratic equation which yields two solutions. The physical meaning of these two solutions is the position of point A and point

B , as point B also satisfies the system of equations. Once the point A and B are determined, the point C and D can be easily determined.

2) Check whether the intersection point is inside the rectangle

There are several methods to check whether a point is inside a rectangle. For example, if the rectangle is defined in a counterclockwise order, the point should lay on the left side of each of the four lines. Another alternative is to calculate the perpendicular distances from the point to all the four lines. Here, we adopt a method using the scalar product of vectors. The condition of the point P_I is inside the rectangle $ABCD$ is

$$\begin{cases} 0 < AP_I \cdot AB < AB \cdot AB \\ 0 < AP_I \cdot AD < AD \cdot AD \end{cases} \quad (4.19)$$

C. Other models

Of course, the human body can be modeled in other ways. For example, it can be modeled as a 3D cuboid with a certain dimension, as shown in Fig. 4.7c. The testing approach is similar to the rectangle model. Instead of checking only one rectangle, six rectangles (cuboid surfaces) need to be examined. Another alternative 3D model is a cylinder.

In this work, we will focus on the plane and the rectangle modeling approaches and validate them experimentally. Later, we can find that these two 2D modeling approaches are sufficient for the camera selection task.

4.4 Camera Selection Based on Quality Metrics

The previous section provides a list of all the available cameras. In this section, a final set of cameras are selected based on a certain defined quality metric. The metric can be defined in various ways, as there are many parameters that affect the performance of the system, such as the angle between the camera's direction and the marker, the distance to the camera, the marker detection error, and the imperfect camera model. From the error model proposed in Section 3.4 and the experimental results in Section 3.7, we can find that the further the marker is from the camera, the bigger uncertainty the results have. Therefore, we chose the distance to the camera as the quality metric. Alternative metrics could be the angle between the marker and the camera or the marker length in the image.

All the available cameras are ranked according to the distance to the camera. The number of finally selected cameras is application-specific and depend on the resources (bandwidth and computational power). It can be easily tuned to adapt to the application.

The aforementioned camera selection procedures are summarized in Fig. 4.8. The complete processing loop is depicted in Fig. 4.9.

FIGURE 4.8: The procedures in the camera selection module.

FIGURE 4.9: The complete processing loop.

4.5 Camera Selection Performance Analysis

The performance of the proposed camera selection approach is evaluated by real tests in three aspects: the model accuracy, the estimation accuracy, and the execution time. The first assessment is designed to validate the accuracy of the proposed occlusion models. The estimation accuracy assessment is to evaluate whether the accuracy of the estimated pose from a selected subset of cameras is acceptable. Finally, the computational cost is compared between the method with camera selection and the method using all the cameras.

4.5.1 Experimental Setup

The experimental deployment and data from the previous chapter are further analyzed for the camera selection performance. Four cameras were deployed in each corner of a room and a user held a tablet walking around in the room. An example of the views from the four cameras is shown in Fig. 4.10. The image resolution was set to 640×360 pixels. In total, 181 frames were captured in the whole trajectory. The images are processed off-line and

FIGURE 4.10: An example of the views from the deployed cameras. (A) Camera 0. (B) Camera 1. (C) Camera 2. (D) Camera 3.

the marker detection results are checked manually. There is no wrong marker detection or missed detection. Therefore, the results can be considered as reference.

4.5.2 Model Accuracy Assessment

As we mentioned in the previous chapter, the results from the Pose Tracker and Marker Tracker are similar. In this chapter, we will use the results from the Pose Tracker for analyzing the camera selection performance. The results from the body plane model, the rectangle model, and the selection without body occlusion modeling are compared to the ground truth, as shown in Fig. 4.11. For the ground truth, “1” means that the camera sees and detects the marker at the time instant and “0” means the camera does not see the marker. For the three camera selection approaches, “1” means the camera is selected and “0” means the camera is not selected.

The green line is the ground truth. That means as long as the marker is completely shown in the camera, it will be detected correctly by the marker detection algorithm. We can see from the figure that camera working modes switch at some points, as the user walked around in the room. The modes are 1-0-1, 1-0, 0-1 and 0-1-0 for the four cameras respectively.

The yellow line represents the result from the selection without occlusion modeling. As long as the camera passes the point-in-view test, it is considered as an available camera. The results from four cameras show clearly that it is not sufficient without modeling the user’s

FIGURE 4.11: Camera selection results. “1” means the camera is active/selected and “0” means the camera is inactive/non-selected. (A) Camera 0. (B) Camera 1. (C) Camera 2. (D) Camera 3.

body, as the device is easily occluded by the user’s body.

The blue line is the result from the plane model. One interesting behavior can be found in the vicinity of the switch points in all the four cameras. There are two switching modes: on-off (1-0) and off-on (0-1). The camera selection method using the plane model attempts to turn off the camera earlier than the ground truth. This is reasonable as the plane model is overestimating the occlusion using a model larger than its actual dimension. For example, in Fig. 4.12a, the marker can be seen by the camera, however in the camera selection procedure, the body plane will occlude the marker. In the switch point off-on, the method attempts to turn on the camera earlier than expected. Two examples are given in Fig. 4.12c and Fig. 4.12d. The reason that the marker detection algorithm did not detect the marker is that the marker is too far from the camera and the marker direction is almost parallel to the camera’s direction.

FIGURE 4.12: Captured images in the vicinity of the switch points. (A) Frame 97 from camera 1. (B) Frame 93 from camera 1. (C) Frame 146 from camera 0. (D) Frame 93 from camera 2.

The red line is the result from the rectangle model. It attempts to turn off the camera a few frames later than the ground truth and turn on the camera earlier than expected. For example, in Fig. 4.12b, the camera is supposed to be able to see the marker according to the rectangle model. However, due to the pointing angle of the tablet, the camera can not see the marker. This is a shortcoming of both models. However, it is too complicated to model this occlusion. The reason that the method attempts to turn on the camera earlier than expected is the same as the one for the plane model.

4.5.3 Estimation Accuracy Assessment

The estimation accuracy with camera selection is assessed by comparing to the results using all the cameras, which is considered as reference. The results are shown in Fig. 4.13. The estimation accuracy with camera selection using the rectangle model is not affected. It is to be expected because the rectangle model tends to select more cameras than the number of cameras that will actually see the marker as above-mentioned. The estimation accuracy with camera selection using the plane model slightly degrades around the moments when the camera hand-off occurs. The average absolute error compared to the reference is 3.2 mm in position estimation and 0.3 degrees in orientation estimation.

FIGURE 4.13: Estimation results from the plane model and the rectangle model compared to the reference. (A) Position estimation results in x , y and z axes. (B) Orientation estimation results in x , y and z axes.

4.5.4 Execution Time Assessment

In order to evaluate the real-time performance, we compare the plane model approach with/without using ROIs and the rectangle model approach with/without ROIs to the method without camera selection (i.e., processing all the images). The five approaches are compared in terms of the execution time. All the tasks are performed in real time in an HP Z420 workstation with an Intel® Xeon® CPU E5-1620 @ 3.60GHz. We processed all the captured 181 frames with the five approaches separately under the same conditions. The execution time for each processing module is recorded and the average execution time for each method and each task is shown in Table. 4.2.

We can find that the camera selection task is very fast, since it consists of algebra operations.

TABLE 4.2: Average execution time for each method. w/o = without and w/ = with.

Approaches	Process all images (s)	Plane model w/o ROI (s)	Plane model w/ ROI (s)	Rectangle model w/o ROI (s)	Rectangle model w/ ROI (s)
Marker detection	0.2456	0.1393	0.0847	0.1497	0.0916
Pose estimation	0.0002	0.0003	0.0002	0.0003	0.0002
Tracking	0.0013	0.0010	0.0009	0.0013	0.0009
Camera selection	Null	0.0003	0.0002	0.0002	0.0003
Total	0.2458	0.1400	0.0860	0.1515	0.0930
Average number of selected cameras	4.000	2.039	2.039	2.182	2.182

As is expected, camera selection without using ROIs reduces the computational cost, as the number of images to be preprocessed is reduced. Besides, the combination of camera selection and generated ROIs largely reduces the total execution time for each loop. On one side, fewer images need to be processed. As is shown in the table, the average number of selected cameras is reduced to around 2 after the camera selection. On the other side, the selected images are not processed entirely. Instead, only a region of interest where the marker would appear are processed. The rectangle model is a bit slower than the plane model as it attempts to process more frames, as we explained earlier.

4.5.5 Discussions

From the experimental results, we can find that without using the proposed occlusion models, all the cameras are supposed to see the marker in the test sequence, as is shown in Fig. 4.11 (yellow lines). However, this is not sufficient to select cameras, as in many cases the marker is occluded by the user's body. By using the proposed occlusion modeling approaches, the average number of selected cameras is reduced to around two with high modeling accuracy. Besides, the estimation accuracy by using the rectangular model is maintained, whereas the approach by using the plane model slightly degrades. The execution time is largely reduced due to the camera selection. The reasons are twofold. On one side, the number of images to be processed is reduced. On the other side, only ROIs are processed, instead of the entire images.

4.6 Conclusions

In this chapter, the camera selection for personal device tracking in multi-camera networks is discussed. Considering the small dimension of personal devices and the usual occlusion, it is a challenging problem. We have presented an approach based on probabilistic techniques, geometric constraints, occlusion modeling, and quality metrics. The pose tracking

results from the previous chapter are used in this chapter to provide the predicted 3D positions of the two reference points. Firstly, a point-in-view test is performed to check whether the predicted reference points lay inside the image, i.e., whether the points are in the field of view of the cameras. Then, the proposed occlusion modeling approaches are applied to the occlusion test. Two occlusion models are proposed: a plane model and a rectangle model. Finally, the available cameras are ranked using the defined quality metric: the distance of the marker to the camera. The performance of the proposed camera selection approaches have been validated by experimental results in terms of selection accuracy and execution time. Experimental results have shown that both the plane model and the rectangle model are appropriate to model the occlusion with high selection accuracy. The plane model overestimates the occlusion and may affect the estimation accuracy, especially when the number of cameras is small. The rectangle model attempts to select more cameras than the actual cameras that will see the marker, as the occlusion model does not take into account the marker's pointing angle. Therefore, the estimation accuracy is maintained. Besides, the computational cost is largely reduced by selecting a set of optimal cameras and generating ROIs to be processed, instead of processing entire images. Especially in a centralized camera network with a large number of cameras, without camera selection, it is hardly possible to provide real-time performance. Therefore, the real-time requirements are met in a multi-camera network using the proposed camera selection approach. All in all, we can conclude that the proposed selection approach is able to provide a real-time seamless tracking of a mobile device with a minimized cost on computation and communication, while maintaining the desired pose estimation quality.

One limitation of this approach is that other obstacles from the environment are not considered, which may also cause occlusions. In the future, the issue about the dynamic obstacles modeling will be studied. Another common limitation of probabilistic based approaches is that the selection accuracy depends on the quality of the predicted information. In this approach, only the predicted state is used. In the future, we will study the possibility of integrating the error covariance matrix of the prediction, as it indicates the uncertainty of the prediction.

Chapter 5

Applications in Smart Spaces

5.1 Introduction

Recently, smart spaces have attracted the attention of industries and researchers from various disciplines, including pervasive and mobile computing, computer vision, sensor networks, human-machine interactions, and wireless networking. Smart space is defined as “one that is able to acquire and apply knowledge about the environment and its inhabitants in order to improve their experience in that environment” in Cook and Das (2007). It integrates different technologies for sensing, reasoning, communication, interaction, identification, and personalization to provide context-aware services and maintain coherent interaction among the heterogeneous smart devices (e.g., controllable lighting and mobile phones). Many applications have been studied by researchers in smart spaces, ranging from navigating through an unfamiliar space (e.g., a museum), to detecting the drowsiness of a driver in automobiles, and to assisting the elderly at home.

For example, in a smart house, any device that uses electricity can be put on the home network and “talk” to each other. Here are some examples of the functions that may be provided by smart houses. Inhabitants may remotely control devices from their couches, offices, hotels, etc.; with the security systems, inhabitants may check the status of the house anywhere and be notified in case of an emergency; the electricity usage may be optimized by turning on only lights where the inhabitant is; the coffee maker may be turned on 10 minutes before the inhabitant gets up; lights, TVs, musics, and windows may be controlled by natural gestures or voice commands; the home may be enabled to learn the preferences of the habitant and provide personalized services. Several prototypes have been constructed in academic fields. For example, the Adaptive House developed by the University of Colorado (Mozer, 2004) is equipped with an array of over 75 sensors which monitor the environmental conditions (e.g., temperature, light, sound, motion, door, and windows openings) and actuators to control the furnace, heaters, lighting units, and ceiling fans. A house named “GatorTech” is constructed by the University of Florida (Helal et al., 2005), which integrates smart appliances, such as smart mailboxes, mirrors, bathrooms, blinds, displays, and ultrasonic location tracking to make a pervasive space. More smart house projects can be found in the review article Alam, Reaz, and Ali (2012). Besides, smart house technologies bring

substantial benefits to the elderly living alone (Demiris, Hensel, et al., 2008). For example, they may notify the inhabitant to take medicine on schedule and send an emergency alert to the hospital when a fall is detected; parents may also check the status of their children from anywhere.

Another usage scenario that is getting popular is smart museums. Visitors can only acquire auditory information from audio guides in traditional museums, whereas multimedia elements, texts, and interactive applications can improve attractiveness. For example, the skeleton of a dinosaur may be layered with muscle, tissue, and skin; when visitors watch a tiger model, they may also hear the roar. Animations or extra texts can help visitors understand better the exhibits. In this context, researchers are making their efforts to make museums “smart” by combining virtual information with the physical world. For example, authors in Bruns et al. (2007) present an enhanced museum guidance system PhoneGuide that uses internal camera and on-device object recognition. Visitors can use their own phones and install the necessary information (application, presentation content, and identification data) from a local base station at the entrance. By taking a photo of the exhibit, the phone will identify the object and play the multimedia content. Therefore, visitors do not need to look up the ID number randomly assigned to the exhibit. In Miyashita et al. (2008), apart from displaying multimedia content, authors add another AR function - route guidance that leads visitors through the exhibition space in a sensible order. In Chang et al. (2014), a mobile AR-guide system is integrated with art appreciation instructions to enhance the learning. The system assists visitors in analyzing the form of the artwork and obtaining the meaning of the images.

In summary, smart spaces are typically composed of embedded sensors, actuators, displays, and computational elements, which are featured in the following aspects (Cook and Das, 2004):

- Remote control of devices. By adding controllers to devices, users may realize remote control of lights, air conditionings, TVs, etc. With this capability, the requirement of physical access to the devices is eliminated.
- Device communication. It is an important component in a network for data sharing, which is largely facilitated with the advances in wireless technologies (e.g., Bluetooth). The communication between devices plays an important role in automation. For example, by sensing the sunlight in a smart house, the angle of solar panels may be automatically adjusted to get optimal energy; when a phone call arrives, the TV may be automatically muted.
- Information acquisition. With the development of sensor technologies, plenty of information can be perceived from the environment, including the temperature and humidity of a monitored area, motion properties of a target, and voices.
- Enhanced services by intelligent devices. Smart spaces can provide users contextual services. For example, the reality can be enhanced by virtual information with the aid

of Augmented Reality technologies; they can be used to monitor the behaviors of the elderly and send an emergency notification to the monitoring center (e.g., a fall); a smart fridge may warn when some food is about to expire.

- Predictive and decision-making capabilities. Another feature of a smart space is the ability to learn the activity patterns and preferences of each user and customize the environment to improve the experience. Based on the sensed data, the system is able to analyze the data and make a certain action to change the state of the environment.

The objective of this chapter is to explore the potential usage of the proposed six-DoF personal device pose tracking approach in smart spaces. The proposed pose estimation module provides the spatial relationship of the mobile device with the world coordinate system. There are many possibilities to exploit this information for smart space services. Firstly, a guideline of integrating the proposed pose tracking approach with smart space services is provided in Section 5.2. Following this guideline, we will present four prototypes of applying the proposed approach to different applications.

Let us imagine a smart space (e.g., home, office or museums) filled with controllable and interactive objects. In order to interact with smart objects, personal devices are preferable considering that a) they are owned as personal devices by most people; b) they are lightweighted and portable; c) they are equipped with a local processor, storage, cameras, sensors, a display screen, and wireless connection abilities. In this context, we will present the approach to use personal devices for pointing applications in Section 5.3. Users are allowed to remotely control the smart devices by pointing at them. The implementation of the application is explained in detail. Smart objects based on Arduino (*Arduino*) and DLNA (*DLNA*) are integrated. Three different user interface designs are presented and compared.

Virtual reality (VR), also known as immersive multimedia, is another application field in smart spaces. It provides a computer-simulated environment and immerses players. Recently, the interest in VR is rapidly grown and several commercial products have been developed, such as Oculus Rift (*Oculus Rift*) and HTC Vive (*HTC Vive*). The immersive environment is achieved by a headset which may cause discomfort for a long time. Instead, by applying the proposed personal device pose tracking approach to VR applications, the users do not need to wear headsets and be isolated from the physical world. The display screen from the device is used to provide the immersive experience. Besides, the motion of the head in the up-to-date products is tracked by inertial sensors. However, without external sources, the absolute position and orientation in the 3D space are not reachable. In the proposed approach, the position and orientation of the device are estimated relative to a world coordinate system. An example of VR applications aimed for immersive learning is implemented in Section 5.4. A virtual 3D garden is built and displayed on the personal device's screen. Users are allowed to walk around to observe the plants from different perspectives.

Section 5.5 is dedicated to describe the methodology to employ the mobile device as a 3D game controller. The movement of the player in the game is controlled by the physical

movements of the user. In this way, it provides an immersive gaming experience. With a full coverage of cameras, the system can be extended to provide 360-degree gaming. A 3D survival game is implemented and integrated with the proposed approach.

The proposed pose tracking approach adopts the outside-in framework. However, it can be converted to an inside-out tracking approach using a similar derivation from the aforementioned monocular pose estimation approach. That is to say, the position and orientation of a personal device are determined by the embedded accelerometers and the proposed marker placed in the environment. The device's internal camera is used for the marker detection task. The novel inside-out pose estimation approach by combining vision and inertial data is applied to augmented books in Section 5.6. The marker is printed in the page border of the book and 3D content is augmented upon a physical book. Compared with the traditional markers, it uses less space in a page and reduces the visual discomfort. Besides, it still works in slightly textures scenes, which is quite challenging for markerless approaches based on scene textures.

In summary, four usage scenarios in smart spaces are explored in this chapter using the proposed pose tracking approach. Besides, it may be adapted to other application fields, such as robot localization and indoor person tracking by attaching the marker on the hats or glasses frames.

5.2 Service Integration

The proposed pose tracking system provides accurate and position and orientation estimation of personal devices, which combines one or more cameras and a paper marker. The server calculates the current pose based on the marker detection results and the measured acceleration data. Then the tracking module smooths the data and in a multi-camera system, the camera selection module guarantees the real-time performance. An example of a visualization service that renders the 3D model of the device and keeps the trajectory of the device is shown in Fig. 5.1.

The smoothed pose is then sent to the device for the application usage. For example, the estimated rotation and translation matrices of the device relative to the world coordinate system, together with the calibration matrix of the tablet camera (which has to be calculated beforehand), allow projecting any 3D point of the real scene on the image screen plane using a perspective transformation. In this way, the objects surrounding the user can be projected into the screen with augmented information. With the aid of the powerful 3D game engine *Unity (Unity)*, a virtual world can be built and connected with the physical world, so that the device can be used to control the virtual player's movement or the perspective of the virtual world.

Here we will give an example of how to integrate the proposed pose estimation system

FIGURE 5.1: The pose estimation system interface. Camera views are displayed. The estimated orientation is represented by the 3D model of the tablet and the position is shown in the room map.

for personal devices into the services. Let us imagine the following application scenario: *“In a museum, a curator wants to augment the exhibits with virtual contents, to be available for visitors through an application installed in their mobile devices. The objective is to provide a better visitor experience, an enhanced presentation of the visual information compared to traditional audio, location-aware guides (e.g., Föckler et al. (2005), Kovavisaruch et al. (2012), Bruns et al. (2007), and Chen, Chang, and Huang (2014)), simpler content selection, and interaction. The service interaction flow is designed to be as follows. On arrival to the exhibition, the museum application will be available for the visitor to be downloaded. Together with the ticket, s/he will receive a colored sticking fiducial that s/he will have to attach to the smartphone/tablet’s border. The visitor will then be ready to enjoy the service. When walking around the exhibition, s/he may feel curious about a painting: just by pointing at it in the way s/he would do it to take a picture, s/he will be able to get distance-dependent virtual information layers that will be immediately superimposed to the real view.”* This augmented reality service requires the visitors’ devices to be very accurately located, both in terms of position and orientation, enabling precise pointing of the target objects. In order to enable the service delivery with minimal cost, the technical solution has to rely on simple infrastructure devices. On the client side, no serious device change can be accomplished, as personal owned devices will be used. All these requirements are met in the proposed system in this thesis.

5.3 Pointing Applications

In smart spaces, everyday objects are allowed to connect to the local network and be controlled by remote command. As is validated in Section 2.7, the proposed pose estimation approach has high degree of accuracy. Therefore, it is suitable for pointing applications. The objective of this section is to implement a prototype of pointing-related applications using personal devices, in which the object to be controlled is selected by pointing at it with a cursor visualized through the screen in the user’s device or an icon. The prototype system

FIGURE 5.2: Arduino Uno and the WiFly shield. (A) Arduino Uno. (B) WiFly.

is composed of the pose estimation module, smart objects, a mobile device, and a user interface. The pose estimation module and the mobile device were explained in the previous chapters. In this following, we will go into detail about the smart objects which include Arduino-based and DLNA-based smart objects, and the user interface.

A. Arduino-based smart objects

Arduino (*Arduino*), as shown in Fig. 5.2a, is an open-source platform and is widely used due to its small size, low cost, and modularity. By connecting to WiFly Shield, as shown in Fig. 5.2b, Arduino is equipped with the ability to connect to a 802.11b/g wireless network, act as a server, and create an ad-hoc network for Internet of Things devices. Data received from the wireless network is forwarded from WiFly to the Arduino, which executes commands in the connected objects. In this way, everyday objects are transformed into smart or interactive objects, being enabled to understand and respond to the environment.

In the prototype, 12 ceiling lamps (as shown in Fig. 5.3a) and one blind (as shown in Fig. 5.3b) with motor are integrated in the network as smart objects using Arduino. Lamps are allowed to be turned on or off by sending commands. The blind is assigned with three functions: go up, stop and go down.

B. DLNA-based smart objects

DLNA stands for “Digital Living Network Alliance” (*DLNA*), which is a standard for sharing media data over a home network among the DLNA-compatible devices. For example, it allows to stream videos from a laptop to the TV or pass the music that is played on a mobile device to a TV. In this prototype, one TV in the entertainment room and one TV in the living room are connected to the local network and work as DLNA clients. Once the DLNA server receives requests from the mobile device, it will deliver the required content (videos, music, or pictures) to the smart TVs. The functions include play, pause, and stop.

(A)

(B)

FIGURE 5.3: Smart lamps and smart blind in the prototyped smart space.

TABLE 5.1: Smart objects and their functions.

Smart objects	Medium	Functions
lamps	Arduino	on/off
blind	Arduino	go up / stop / go down
TVs	DLNA	play/pause/stop images/videos/audios

The smart objects and their functions are summarized in Table 5.1. A realistic spatial deployment of the objects in a smart space are depicted in Fig. 5.4.

C. Graphic User Interface (GUI) design

The GUI is implemented in a mobile device screen, enclosed by four parts: image view, setting, information, and reaction, as illustrated in Fig. 5.5a. The image view is to display

FIGURE 5.4: The floor map of the Smart room prototype and the deployment of smart objects. (A) Floor map. (B) A 3D snapshot.

real-time images from the device's internal camera. Three operating modes have been implemented. Users are allowed to switch from one mode to the other by operating in the setting panel. The information panel includes the estimated position, orientation, and the measured gravitational acceleration. The reaction panel contains the icons for each smart object and the available operational commands.

The first operating mode is to augment all the objects in view with labels, as is shown in Fig. 5.5b. Users can achieve command control of the objects by tapping the augmented icon labels. The second operating mode is to use a crosshair, which is centered in the middle of the image view with a predefined radius for users to target the objects that they are interested in, as shown in Fig. 5.5c. When the object is in view, on the right part of the panel, the corresponding image icon will flash for a few seconds for the user to take action. When the TV icons are selected, an extra panel with the available multimedia resources on the DLNA server appears for the user to choose. The third mode is similar to the second one, but instead of showing all the icons, only the ID of the targeted object is shown in the circular icon, as shown in Fig. 5.5d.

D. System integration

In the first place, the PC server estimates the position and orientation of the mobile device and sends the results to the device. The locations of the smart objects are measured beforehand. Based on the estimated pose and the calibrated device camera, the 3D objects are projected into the 2D image. Users can remotely control the objects by tapping the augmented icons or the flashing icons in the GUI. The complete system architecture is depicted in Fig. 5.6.

E. Discussions

We collected the feedback from users about the three user interface designs. The first design has several disadvantages. Firstly, in case of a large number of smart objects in a room, there may be too much information in the image if many icons are painted in the image.

FIGURE 5.5: Graphic User Interface design. (A) The GUI composition. (B) A snapshot of the operating mode to augment all the objects in view. (C) A snapshot of the operating mode to flash only objects in the crosshair. (D) A snapshot of the operating mode to show only the ID of the objects in the crosshair.

FIGURE 5.6: System architecture in the smart space prototype.

Secondly, in order to achieve smooth results, users have to hold the mobile device stably. It may be difficult to maintain the posture when users have to move one hand to tag an icon in the image. Furthermore, there is unavoidable jitter from the pose estimation results (even though they are smoothed), as users may have small movements when targeting the tablet to a certain object. Since human eyes are sensitive to the registration error, the jitter may be unpleasant for users. The advantage of using a crosshair is that the interface is simpler. Besides, as long as the projected smart object falls inside the defined circle, it is considered as targeted, which is less sensitive to the registration error. The simplest user interface design is the third design. However, users have to remember the objects that are integrated in the service. Instead, in the second design, users are aware of smart objects from the list of icons on the panel.

FIGURE 5.7: A systematic description of the VR application.

5.4 Virtual Reality for Immersive Learning

Virtual reality (VR) is a technology which allows users to completely immerse in a virtual environment and interact with the digital contents. The virtual environment is constructed by a variety of techniques that respond to the user's input (e.g., gesture, voice command, etc.). VR has great potential in the field of education and training. For example, it can be used in flight simulation for pilot training (Aslandere, Dreyer, and Pankratz, 2015). It can also be used for surgical simulation to train skills in an operating room environment (Peterson and Robertson, 2013).

In this section, we have applied our proposed handheld device pose estimation approach to VR for immersive learning based on Unity (*Unity*). The learning situation differentiates from traditional learning materials on its ability to provide learners a realistic 3D scenarios and an immersive experience. The concept of the application is that a user is enabled to navigate in a virtual world with his own physical movement. A smart space is monitored by multiple cameras to track the mobile device, as shown in Fig. 5.7. A user can hold the mobile device with an installed application and walk around in the scenario. From the device screen, the user will see the models of the virtual world. Besides, the user is enabled to observe the virtual world from different perspectives based on the estimated position and orientation of the device. In this way, a VR application providing immersive experiences is achieved with the aid of the proposed pose estimation system.

The innovation of this approach is that users do not need to wear headsets so that they are still aware of the physical world while immersing in the virtual world. Compared with the markerless approaches, the method works in a slightly textured scene (e.g., floor or table), which is quite challenging for markerless approaches based on scene textures.

A. Hardware and software

Unity (*Unity*) is a cross-platform game engine which is widely applied to develop video

games for PC, consoles and mobile devices. In this application, a virtual 3D garden is constructed based on Unity. There are two programs running simultaneously: a) the pose estimation module runs on the server side and outputs the position and orientation of the device relative to a certain coordinate system; b) the mobile device runs a Unity program which is capable to collect the accelerometer measurements and send them to the server for the pose calculation. Besides, once the estimated pose is received, it changes the view of the virtual garden and displays it in the personal device's screen.

B. Virtual garden modeling

A virtual 3D botanical garden is constructed by the powerful game engineer - Unity with various plants, including cactus, palm trees, flowers, and mushrooms. The models of each object are fitted to realistic sizes with refined details. Moreover, each plant is labeled with its name for learning purpose. An example of the virtual world is shown in Fig. 5.8.

FIGURE 5.8: An example of the virtual botanical garden.

C. Virtual camera controlled by personal devices

In Unity, there is a virtual camera which can be placed in a certain position with a certain direction. When the Unity program starts, it will show the content that is viewed by the virtual camera. To give a realistic and immersive feeling, the system adjusts the position and direction of the virtual camera according to the pose of the actual camera of the personal device. The effect is that the virtual camera substitutes the real camera with perspective views of a virtual garden.

Firstly, the effective area has to be defined depending on the scale of the virtual world against the real world. Then, the pose of the mobile device is converted to the pose of the virtual camera in the garden. The view from the virtual camera is then displayed in the device's screen. In this way, users can get a realistic feeling of being in a physical garden and learn about the plants, whereas they are wandering through a virtual world.

5.5 3D Gaming Based on Unity

The video game industry has become a large market aimed at entertainment, education, physical training, etc. Various game controllers have been recently introduced, such as handheld thumb joystick game controllers, Nintendo® Wii Remote (*Nintendo Wii Remote*), Sony PlayStation® Move (*Sony PlayStation*), and Microsoft® Xbox Kinect (*Kinect*). However, it remains an open challenge to improve the user experience and interaction with games. Due to the booming of the mobile industry nowadays, most of mobile devices are equipped with multi-sensors, touch-screen, and wireless connection abilities. These benefits make mobile platforms favorable to develop interactive games on.

In this section, we present how to use the pose estimation system proposed in this thesis to enable mobile devices as six degree-of-freedom (DoF) video game controllers. The proposed game controller is featured in the following aspects:

- **Immersive gaming experience:** conventional game controllers use button-presses to control the movements of the player in the game. The proposed approach uses the physical movements of the mobile device as input for the game. Thus this design allows a more immersive gaming experience, letting the user feel that they are the players in the game instead of controlling the players.
- **3D orientation detection:** the built-in accelerometer senses the inclination, as other accelerometer based game controllers do. Additionally, the proposed system can detect the yaw angle with the aid of fiducial markers. Thus the controller can be used as a pointing device.
- **360-degree gaming experience:** with three or more cameras, it is possible to cover a 360-degree gaming area. To our knowledge, it is the first 360-degree gaming experience without the need of wearing a virtual reality headset.
- **Alternative display device:** the game can be alternatively displayed on the screen of the mobile device, which gains the sense of immersion.
- **Low-cost, accurate, and fast:** the proposed system supports off-the-shelf webcams, laptops and any mobile device equipped with a 3-axis accelerometer. The position and orientation estimation is accurate and fast, meeting the requirements of games.

The pose estimation module plays an important role in the immersive game, as the movement of the player in the game is controlled by the physical movement of the user. Together with the powerful game engine Unity, an immersive 3D game applying the mobile device as a game controller has been developed, which is able to provide an immersive 360-degree game experience.

A. Hardware and software

In this game, Unity is used to construct the 3D scene and control the actions of the models through C# scripts. Three programs run simultaneously: a) the mobile device measures the acceleration and outputs the measurements to the pose estimation module; b) the pose estimation module, programmed in C++, runs on the server side and works with one or more cameras. It outputs the estimated position and orientation of the device relative to the world coordinate system; c) the Unity program, programmed in C#, runs on the server side and can be alternatively displayed on the device's screen via Unity Remote to provide more possibilities for multi-media elements integration. It interprets the received pose to control the actions of the player. The communication among them is via socket.

B. Game scenario

The game is designed as an immersive 3D shooting survival game. The environment is set as a room with several 3D models, such as chairs, closets and toys, as shown in Fig. 5.9a. Three different enemies are included and they are set to follow the player. The objective is to shoot the enemies and avoid them getting close to the player. The player can shoot with the gun but has a limited amount of bullets. The player and enemies are restricted to move only in the floor plane. A screenshot of the game is shown in Fig. 5.9b.

FIGURE 5.9: Game scenario. (A) The environment setup. (B) A screenshot of the game.

C. Virtual content control by personal devices

1) Association between the real world coordinate system and the virtual world coordinate system

The 3D virtual space is created by Unity based on its definition of coordinate system and scale, whereas the real world has another coordinate system. It is necessary to associate the two coordinate systems in order to control the movements of the player in the game through the motion of the device. In this prototype, the horizontal size of the virtual room is 50×50 Unity unit and the coordinate system is left handed, where x positive is right, y positive is

FIGURE 5.10: The setup of cameras and the gaming area.

FIGURE 5.11: The game areas. (A) The effective game area in the real world (an example of 3 m×3 m). (B) The virtual game area of 50×50 Unity unit.

up, and z positive is into the screen. The corresponding effective area in real world can be defined on the ground which is under the coverage of the cameras, as shown in Fig. 5.10 (red rectangle). The origins of both systems are located in the center of the game area, as is shown in Fig. 5.11. The scale conversion can be easily obtained with known dimension of both game areas. For example, in Fig. 5.11, the scale is 50:3 m. Alternatively, the game can be projected in the wall, as shown in Fig. 5.10.

2) Player movement control

The user holds the tablet and walks around in the effective game area to control the position of the player. The effect is when the user moves to one direction, the game player is also proportionally moved to the same direction. The estimated position of the hand-held device relative to the coordinate system shown in Fig. 5.11a is converted to the coordinates in the Unity coordinates based on the known scale and axes conversion. The player is then moved

FIGURE 5.12: Direction control of the player. From (A) to (B) is the horizontal direction change. From (A) to (C) is the vertical direction change.

to the calculated position. The command is "move($x \cdot \text{scaleX}$, 0, $y \cdot \text{scaleY}$)", where (x , y) are the coordinates in real world and (scaleX , scaleY) are the scales of virtual coordinates against the world coordinates in X and Y axes. This design allows users to have a feeling that they are the players in the game. Therefore, the immersion is gained in the game.

3) Gun direction control

The horizontal and vertical direction of the gun are determined by the yaw and pitch rotation angles of the controller respectively, as shown in Fig. 5.12. As the amount of bullets is limited, the pitch angle is also designed as a shooting trigger. When the device is pointed downwards, it starts shooting, and when the device is pointed upwards, it stops shooting.

5.6 Augmented Books

Given the exciting development and functionality of AR technologies, researchers believe that AR is beneficial to education and training (Billinghurst, Kato, and Poupyrev, 2001; Wu et al., 2013). One specific emerging application field in education is AR books. According to a study by authors in Marshall (2005), readers still love the physical printed books, even though the electronic books are becoming popular recently. This fact invoked a research study of applying AR technologies to digitally enhance and augment traditional real books. An observational study implemented by Dünser and Hornecker (2007) found that AR books are attractive to users and can enhance the learning process thanks to the augmented multimedia content, such as videos and animations. As an example, "MagicBook" is presented in Billinghurst, Kato, and Poupyrev (2001), which allows users to see animated 3D content using ARToolkit markers. One obvious drawback of these marker-based approaches applied to books is that markers have to be big enough to provide accurate results, thus they take up a certain amount of space in the page. This largely influences the visual effect or aesthetic. A hybrid visual tracking approach is proposed in Yang et al. (2008), which merges the merits of fiducial marker-based tracking and markerless tracking. They use a tiny marker for page

FIGURE 5.13: The world coordinate system in augmented books.

recognition and a random forest for pose estimation. However, no accuracy assessment is provided.

In this section, the proposed outside-in tracking approach to combine accelerometer measurements and a colored marker is converted into an inside-out tracking framework. That is to say, the marker is deployed in the environment and observed by the device's internal camera. The pose of the device relative to the marker is retrieved by fusing the marker detection result and the accelerometer measurements. The advantage of this approach is that no external camera is required. This framework can be applied to an emerging application field - augmented books. Multimedia elements (e.g., text, images, sound, and video) are integrated in physical real books to enhance the learning process or provide amusement and interaction. Compared to the traditional square markers, the proposed thin marker can be printed on the border of a page, which causes much less visual discomfort. The marker detection algorithm is the same as we mentioned in Algorithm 1. The data fusion approach is different from the previous approaches and it is explained in Section 5.6.1. Besides, the performance of the proposed approach is evaluated experimentally in Section 5.6.2.

5.6.1 System Description

The position and orientation of the mobile device relative to the book are vital to augmented correctly the virtual content. The world coordinate system in this application (depicted in Fig. 5.13) is defined as: a) its origin is located in the reference point P_1 ; b) its X-axis is aligned with the marker's longitudinal direction; c) its Y-axis is perpendicular to the X-axis and aligned with the page border pointing up; d) its Z-axis is perpendicular to the X-axis and Y-axis, following the right-hand rule. The camera coordinate system is identical as explained before. The pose estimation of the device is achieved by a fusion of accelerometer measurements from the mobile device and the marker detection results. The algorithm is explained in detail in Appendix A.

FIGURE 5.14: An example of a 3D virtual model popping up out of a page using the proposed approach.

Texts and planar images may be abstract for learners to understand or imagine in spatial domain. However, with the aid of AR technologies to superimpose supplementary information rendered in a multimedia format, learning or reading can become easier and more attractive than using a physical book. For example, AR can enable readers to visualize a world globe upon a 2D world map or text. Also, it can enhance the learning process of chemistry, physiology, astronomy, etc. in an immersive and interactive way. A variety of multimedia elements can be integrated in a physical book:

- 2D content: texts, images, illustrations, hyperlinks, videos, and animations. For example, an animation of how to write alphabets is helpful for preschool children.
- 3D content: 3D models and animations. In Fig. 5.14, we give an example of using the proposed approach to pop up a 3D tiger model out of a text related to tigers. Readers can also rotate the book or the display device to have a 360-degree view of the model.
- Interaction: users are allowed to interact with the physical world through manipulating the virtual content. For example, users can move and place an interactive character to accomplish a game.
- Sound: music and audio guide.

5.6.2 Performance Analysis

In this section, the accuracy and the real-time performance of the propose approach are assessed.

FIGURE 5.15: (A) Accuracy assessment using the corner detection results from OpenCV as ground-truth. Red circles are the calculated corners positions and the green circles are the corner detection results from OpenCV. The yellow circles are detected reference points positions in the image. (B) The world coordinate system is projected in the image based on the estimated camera pose.

A. Testing procedure

A tablet PC Nexus 10 running on the Android operating system is chosen as the AR device. The device camera's intrinsic parameters are calibrated beforehand with the image resolution set to 640×480 pixels. The marker is designed as $19 \text{ cm} \times 0.3 \text{ cm}$ (yellow: 2.5 cm, magenta: 14 cm and cyan: 2.5 cm), which is a suitable size for an A4 page. The pose estimation accuracy is estimated dynamically in terms of projection errors in pixels, i.e., the displacement of the augmented object from the real object in the image. We benchmark our results to the ground truth generated by OpenCV library (*OpenCV*), an open source, widely-used, and cross-platform computer vision library. OpenCV finds the position of internal corners of a chessboard using the function *findChessboardCorners()*.

To set this benchmark up, a chessboard composed of 4×3 grids of $20 \text{ mm} \times 20 \text{ mm}$ has been chosen as the physical augmented object (forming 12 corners). The chessboard is placed in a known position in the global coordinate system, as shown in Fig. 5.15a. The detected positions of these 12 corners by OpenCV are considered as ground truth. In our proposed system, firstly, the position and the orientation of the tablet camera relative to the stripe marker are estimated by fusing vision and inertial data. Secondly, based on the estimated pose, the 12 corners with known world coordinates are projected to the image. Then, the root-mean-square (RMS) of the displacements (in pixels) of the calculated positions of the corners from ground truth are considered as the average projection error in the current frame.

In Fig. 5.15, two captured examples of the running system are presented. The comparison of the proposed system to the OpenCV chessboard corner detection is shown in Fig. 5.15a. As we can see, the projected chessboard corners using our approach are in close correspondence to the detected chessboard corners. Based on the estimated camera pose, the origin and X-Y-Z axes of the world coordinate system are projected in the image, as shown in Fig. 5.15b.

FIGURE 5.16: RMS errors for 400 frames with varying camera motion and varying distance.

The RMS errors for 400 frames with varying camera motions and varying distances to the marker are illustrated in Fig. 5.16. Then, we calculated the overall RMS error of these 400 frames and obtained an average projection error of 4.8 pixels. This error is insignificant for most of the applications.

B. Execution time

Regarding the execution time, an evaluation was carried out by computing the average time needed for frame capturing, image processing, pose calculation, and image visualization. The OpenCV chessboard detection method (in particular, the function *findChessboardCorners()*) has two operating modes: a fast check mode and an adaptive thresholding mode. The fast check mode runs a fast check on the image to look for chessboard corners and shortcuts the call if none is found. It drastically speeds up when no chessboard is observed. The adaptive thresholding mode is robust to the illumination changes. However, it is time-consuming. We compared the execution time of the proposed system with the OpenCV method both in the fast check mode and the adaptive thresholding mode.

During the experiment, we implemented the three methods on the same device and calculated the average processing time for 1200 frames. Both the proposed system and the reference system in the fast check mode have achieved 11 frames/s, while the reference system in the adaptive thresholding mode has achieved 2.5 frames/s.

C. Some notes

Although in this prototype page determination is not implemented, it can be achieved by some already existed approaches, such as QR codes (Kan, Teng, and Chou, 2009), number recognition using optical character recognition (Qadri and Asif, 2009), or the tiny marker

proposed in Yang et al. (2008). Page determination is left as future work. Furthermore, an AR book with augmented content and interaction will be completed and related issues such as UI design and content rendering will be studied.

5.7 Conclusions

In this chapter, we have studied the usage of the proposed pose tracking approach for personal devices in smart spaces. Four prototypes have been implemented and analyzed, including pointing applications, VR for immersive learning, 3D gaming and, augmented books. In pointing applications, smart objects based on Arduino and DLNA are integrated in the service for remote control. Three user interface designs are proposed and compared. Then, personal devices are applied to VR applications to substitute wearable headsets. Besides, the absolute position and orientation in the world coordinate system are provided by using our system. Furthermore, personal devices are converted to 3D game controllers, which manipulate the movements of players in the game by users' physical movements. Finally, the proposed approach is transformed for inside-out tracking, which is then applied to augmented books. By printing the marker in the border of a page, the visual discomfort is largely reduced. Besides, it works for slightly textured pages. As is validated in the previous chapters, the approach to pose estimation is simple, accuracy and fast. Therefore, it fulfills the requirements for applications in smart spaces.

In summary, the proposed six-DoF pose tracking approach for hand-held devices is a general solution which can be applied to various application fields in the domain of smart spaces, such as indoor AR, VR, person and object tracking, robot localization, and pointing-related applications. Thanks to the thinness of the marker, it is easy to be attached to a small surface. Apart from mobile devices, it can also be attached to a robot for robot localization or glasses frames for person tracking, or the wings of a drone for indoor drone tracking.

In the future, the performance of the designed prototypes may be evaluated comprehensively by user experience assessment. Besides, the system may be adapted for other application fields, such as robotics and person tracking.

Chapter 6

Conclusions and Future Research Directions

This chapter summarizes the whole work of the thesis and discusses the main contributions and results. Besides, the limitations of the current work are pointed out and future research directions are suggested.

6.1 Conclusions

In this thesis, the general problem of 6-DoF pose estimation and tracking for indoor environment has been studied. The main contribution is to present a handheld 6-DoF pose tracking system, which encloses four modules: the pose estimation module, the tracking module, the camera selection module, and the application module. They are summarized in the following.

Pose estimation module. A hybrid handheld pose estimation system has been proposed by combining vision and inertial data. The system is based on a colored fiducial stuck on the device, embedded accelerometers in the handheld device, and one or more external vision sensors. Several multi-sensor data fusion techniques for pose estimation have been proposed: monocular vision-based, stereo vision-based, multiple vision-based, and occlusion handling. The monocular vision system, although having reduced accuracy, has lower constraints on the density of the camera deployment. The performance of the system has been validated in terms of accuracy and execution time. In the accuracy assessment, the proposed system has been compared to the ground truth and a reference marker-based system. Experimental results show that the proposed system has achieved an accuracy in the order of centimeters for the position estimation and few degrees for the orientation estimation, working at around 10 frames/sec in a stereo-vision system, which fulfills the real-time requirements of most applications. Compared with the reference system, the proposed system provides longer working range and more stable results. Furthermore, the proposed marker surpasses the marker from the reference system in terms of dimension and shape. The thinness allows it to be attached to a small surface such as borders of mobile devices, robots, or

eyeglasses frames, whereas traditional markers need a big and flat surface to be placed. The integration of accelerometers reduces the complexity of the vision system, as it provides the inclination information. Additionally, the system is easy to deploy, requiring a minimum of setup and configuration to get running. On the whole, we can summarize that the system is a low-cost, accuracy, fast, robust, easy to deploy, and general solution for handheld pose estimation.

Tracking module. Two tracking algorithms based on the state-space model and probability have been presented, which are named as Pose Tracker and Marker Tracker. The Pose Tracker filters directly the pose and the Marker Tracker filters the estimated 3D positions of the two reference points. A complete system error model has been analytically derived based on the error propagation rules and then validated experimentally. On one side, the error model is a useful tool to evaluate the stability of the system at any given position and any given orientation in the working range. On the other side, it provides an estimate of the measurement error covariance matrix as close as possible to the actual values, which is an important factor to the performance of the filters. The design and implementation of the filters have been explained in detail, including the filter initialization, parameter settings, dynamic system model, measurement, and process noise model. The consistency of both filters has been tested by analyzing the innovation sequence from real tests. Both filters have been validated to be consistent. Besides, the innovation sequence has been exploited to detect outliers to avoid abrupt changes in the output. A special measure has been taken to deal with the false detection of outliers due to the switch of measurement sources (cameras), which has been validated to largely reduce the false detection due to this fact. Moreover, the predicted state of the system from the filter has been used to obtain a region of interest of the image at the next time instant to be processed, instead of the entire image. This procedure largely reduces the computational cost, as only a certain part of the image is processed. Also, it improves the system robustness, as noise or other objects outside the region that may cause confusion will not influence or even disturb the processing. The two filters are used to smooth the noisy raw estimation. The Pose Tracker may improve the position and orientation estimation accuracy. One disadvantage of the Marker Tracker is that the filtered estimation of the yaw rotation angle may degrade, as it is calculated by the filtered relative positions of the two reference points, which is sensitive to small changes.

Camera selection module. A camera selection mechanism in a multi-camera network has been presented, which is based on the predicted state of the system, occlusion modeling, and quality metrics. In the first place, a subset of cameras that will have the marker in view at the next time instant is selected based on the point-in-view test and the occlusion test. Several modeling approaches are proposed for the occlusion test. Among them, the plane model and the rectangle model are explained in detail and implemented. In the second step, a quality metric is defined by the distance of the device to the camera to select an optimal camera set from the results obtained from the first step. The performance of these two models have been evaluated by real tests and compared to the ground truth. Experimental results show

that both models are able to select cameras with high accuracy, while the plane model overestimates the occlusions leading to some missed detection and the rectangle model attempts to select a bigger set of cameras than the set that really “see” the marker, especially in a large camera network. Furthermore, the computational cost is largely reduced by the camera selection procedure, especially for a large camera network. Therefore, we can conclude that the proposed selection mechanism is able to provide a real-time seamless tracking of a mobile device with reduced cost on computation and communication, while maintaining a desired quality.

Application module. The proposed 6-DoF pose tracking approach for handheld devices is a general solution which has potential in several application fields in the domain of smart spaces. In this thesis, we have explored several usage scenarios of the proposed system. Firstly, a service integration guideline is provided. Then, following the guidelines, it has been applied to achieve one main function of smart spaces - remote control. By pointing the device to the smart devices (e.g., lamps and TVs), users are enabled to control the devices remotely without the need of physical access to them. It is also a potential solution for the active research field of VR. An application for immersive learning using the proposed approach has been implemented to exemplify its possibility in VR. Besides, the potential of the proposed system in the field of immersive gaming has been explored. Together with the powerful game engine Unity, an immersive 3D game applying the mobile device as a game controller has been developed, which is able to provide an immersive 360-degree game experience, as the physical movement of the mobile device/user is handled as the game input to control the actions of the player. Furthermore, the proposed monocular vision system based on the outside-in tracking structure has been transformed into an inside-out tracking approach, that is, the pose of the device is retrieved by detecting the marker attached in the environment through the device’s internal camera. It has been then applied to AR applications to augment physical books with virtual content. Experimental results have shown that by simply printing the colored marker in the border of a page, the spatial relationship between the mobile device and the marker can be estimated accurately and fast. Apart from tracking mobile devices, the proposed system is also capable to perform other tasks, such as tracking person and object (e.g., for warehouse analytics, indoor drone tracking, or activity assessment) and robot localization (e.g., for industrial applications or autonomous robot navigation). All in all, we can claim that the proposed system has rich potential for services in smart spaces.

6.2 Future Research Directions

In the following, we summarize the limitations of the proposed system and suggest some future research directions and possible improvements.

Multi-user tracking. Regarding the proposed pose estimation system, one main limitation is that the current system is only able to track one user. In case of multiple users, each device

has a unique identification labeled by its IP address. It is necessary, however challenging to distinguish them from the images. To address the single user limitation, several potential approaches may be explored. For example, one simple solution is to assign different markers to different devices. However, in this case, the number of users may be limited to a small amount. Alternatively, each device may be tracked by the proposed filter in 3D space and distinguished by their trajectories. Data association techniques may be studied to determine the best match based on the tracking results. Additionally, the appearance of each user encloses rich information, which may be used for identification (e.g., color histogram).

Optimization of the marker detection algorithm. The pose estimation accuracy relies heavily on the marker detection accuracy. There are several factors that may cause inaccuracy in the marker detection. For example, similar colors in the surrounding area may cause confusion. In varying illumination conditions, the fixed thresholding method for color segmentation may be insufficient. Therefore, the marker detection algorithm may be improved in robustness against the environmental noise and the lighting. One possible solution is to set the threshold adaptively.

A multi-camera network with a large number of cameras. The current multi-camera network is composed of four webcams, which are connected to a server through USB cables. In the case of a network with a large number of cameras, it is not practical to connect them to a single server due to the limited bandwidth in the USB communication. Therefore, potential solutions may be to increase the number of servers or adopt smart camera nodes. In any case, a main issue has to be taken into account: synchronization. Sources from different devices must be synchronized before data fusion. Another issue is the optimization of the data transmission among devices to minimize the communication cost. Moreover, the proposed camera selection algorithm may be even more thoroughly analyzed in a large camera network, whereas the current deployment with four cameras is limited to a room-size scenario.

Environmental occlusion modeling. In the current camera selection approach, only the user's body is considered and modeled as occlusion, whereas other static and dynamic obstacles from the environment are not considered in the scope of this thesis. It may improve the selection quality by modeling them in a realistic scenario.

Usage of prediction uncertainty. The predicted covariance of the estimate provided by the Kalman filter is a piece of useful information, as it indicates the uncertainty of the prediction. It may be exploited to enhance the camera selection algorithm, as only the predicted state is used in the current algorithm.

User experience testing. The user experience testing plays a significant role in a system development, as it collects the feedback from users about a set of specific measures, such as the utility, usability, aesthetics, and value. Therefore, it is still a pending task to conduct a user experience assessment with regard to the proposed applications.

Other application fields. Several prototypes in different application fields based on the proposed pose tracking system have been implemented in this work. Still, there is much

room to explore promising use cases to leverage the proposed system in various fields, such as robotic, Internet of things, AR and VR.

Fusion of outside-in and inside-out tracking. The proposed outside-in tracking provides accurate results, however it requires the line-of-sight of cameras. Besides, small errors in the pose estimation using external cameras may be propagated to big errors in the end applications (e.g., errors in the world objects' projections on the device's image plane). On the other side, the inside-out markerless tracking approach uses the device's internal camera to observe the environment and interprets the relation between the device and the world through tracking natural features in the image sequences. Therefore, it does not have coverage problem. However, it needs a starting point/ state and the accuracy may decline with time. Therefore, we would like to suggest a research direction, that is, the fusion of outside-in tracking and inside-out markerless tracking to overcome the limitations of each approach. The fusion of the two approaches may improve the accuracy and robustness of the system. Besides, it may extend the system to a large-scale scenario with a small amount of cameras.

Appendix A

Pose Estimation for Augmented Books

This section gives mathematical derivations of 6-DoF pose from the marker detection results and the accelerometer measurements.

Let us start with the detected reference points from the images. A pinhole camera model is adopted and the mapping of 3D points in world coordinates to 2D points in the image plane can be mathematically described as

$$\lambda \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = K(R_{wa}\mathbf{P} + \mathbf{t}_{wa}) = \begin{bmatrix} f_x & 0 & u_0 \\ 0 & f_y & v_0 \\ 0 & 0 & 1 \end{bmatrix} \left(\begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \\ t_z \end{bmatrix} \right), \quad (\text{A.1})$$

where λ is a scale factor; (u, v) are the coordinates of a 2D point; K is a matrix containing camera intrinsic parameters: the focus length f_x, f_y and the principal point (u_0, v_0) ; R_{wa} is the rotation matrix from the world coordinate system to the accelerometer coordinate system; $\mathbf{t}_{wa}$ is the translational vector.

Firstly, we apply Eq. A.1 to the reference point $\mathbf{P}_1$, whose coordinates in the world coordinate system are $(0, 0, 0)^T$ and whose coordinates in the camera frame coordinate system are (u_1, v_1) and get

$$\lambda_1 \begin{bmatrix} u_1 \\ v_1 \\ 1 \end{bmatrix} = K(R_{wa} \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix} + \mathbf{t}_{wa}) = K\mathbf{t}_{wa} = \begin{bmatrix} f_x t_x + u_0 t_z \\ f_y t_y + v_0 t_z \\ t_z \end{bmatrix}. \quad (\text{A.2})$$

The reference point $\mathbf{P}_2$ is expressed in the world coordinate system as $(D, 0, 0)^T$, where D is a known value indicating the distance between two reference points. Its 2D position in the image is indicated as (u_2, v_2) . We apply Eq. A.1 to the reference point $\mathbf{P}_2$ and get

$$\lambda_2 \begin{bmatrix} u_2 \\ v_2 \\ 1 \end{bmatrix} = K(R_{wa} \begin{bmatrix} D \\ 0 \\ 0 \end{bmatrix} + \mathbf{t}_{wa}). \quad (\text{A.3})$$

Please remind that the pitch and roll rotation angles from the rotation matrix R can be deduced by accelerometer measurements in Eq. 2.13 and Eq. 2.14. Therefore, in the system of equations composed by Eq. A.2 and Eq. A.3, there are six equations with six unknown variables, i.e., λ_1 , λ_2 , the yaw rotation angle and the translational vector $\mathbf{t}$ (3 elements). As a consequence, all the parameters of the six-DoF pose can be resolved from this determined system of equations.

Bibliography

- Abowd, Gregory D et al. (1999). "Towards a better understanding of context and context-awareness". In: *Handheld and ubiquitous computing*. Springer, pp. 304–307.
- Aggarwal, P et al. (2008). "A standard testing and calibration procedure for low cost MEMS inertial sensors and units". In: *Journal of navigation* 61.02, pp. 323–336.
- Alam, Muhammad Raisul, Mamun Bin Ibne Reaz, and Mohd Alauddin Mohd Ali (2012). "A review of smart homes - Past, present, and future". In: *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on* 42.6, pp. 1190–1203.
- Arduino. <https://www.arduino.cc/>.
- ARToolKit. <http://www.hitl.washington.edu/artoolkit/>.
- ARTrack. <http://www.ar-tracking.com/>.
- Aslandere, Turgay, Daniel Dreyer, and Frieder Pankratz (2015). "Virtual hand-button interaction in a generic virtual reality flight simulator". In: *Aerospace Conference, 2015 IEEE*. IEEE, pp. 1–8.
- Azuma, Ronald and Mark Ward (1991). "Space resection by collinearity Mathematics behind the optical ceiling head-tracker". In: L95–L98. *Department of Mathematics and Statistics, University of Edinburgh, King's Buildings, Mayfield Road, Edinburgh EH9 3JZ, Scotland*. E-mail address: davidmjc@maths.ed.ac.uk *Department of Mathematics and Computer Science, SDU-Odense University, Campus*. CiteSeer.
- Baghyasree, T, K Janakiraman, and D Parkavi (2014). "Efficient remote video monitoring for industrial environment based on LPC2148 and GSM". In: *Indian Journal of Science and Technology* 7.9, pp. 1333–1341.
- Bahl, Paramvir, Venkata N Padmanabhan, and Anand Balachandran (2000). *Enhancements to the RADAR user location and tracking system*. Tech. rep. technical report, Microsoft Research.
- Bar-Shalom, Yaakov (1987). *Tracking and data association*. Academic Press Professional, Inc.
- Bar-Shalom, Yaakov, X Rong Li, and Thiagalingam Kirubarajan (2004). *Estimation with applications to tracking and navigation: theory algorithms and software*. John Wiley & Sons.
- Bar-Shalom, Yaakov, Peter K Willett, and Xin Tian (2011). "Tracking and data fusion". In: *A Handbook of Algorithms*. Yaakov Bar-Shalom.
- Bay, Herbert, Tinne Tuytelaars, and Luc Van Gool (2006). "Surf: Speeded up robust features". In: *Computer vision—ECCV 2006*. Springer, pp. 404–417.
- Billinghurst, Mark, Hirokazu Kato, and Ivan Poupyrev (2001). "The magicbook-moving seamlessly between reality and virtuality". In: *Computer Graphics and Applications, IEEE* 21.3, pp. 6–8.

- Bird, Jeff and Dale Arden (2011). "Indoor navigation with foot-mounted strapdown inertial navigation and magnetic sensors [emerging opportunities for localization and tracking]". In: *Wireless Communications, IEEE* 18.2, pp. 28–35.
- Blom, HP (1984). "An efficient filter for abruptly changing systems". In: *The 23rd IEEE Conference on Decision and Control*. 23, pp. 656–658.
- Bruns, Erich et al. (2007). "Enabling mobile phones to support large-scale museum guidance". In: *IEEE multimedia* 14.2, pp. 16–25.
- Chang, Kuo-En et al. (2014). "Development and behavioral pattern analysis of a mobile guide system with augmented reality for painting appreciation instruction in an art museum". In: *Computers & Education* 71, pp. 185–197.
- Chaudhary, Ankit et al. (2013). "Intelligent approaches to interact with machines using hand gesture recognition in natural way: a survey". In: *arXiv preprint arXiv:1303.2292*.
- Chen, Chia-Yen, Bao Rong Chang, and Po-Sen Huang (2014). "Multimedia augmented reality information system for museum guidance". In: *Personal and ubiquitous computing* 18.2, pp. 315–322.
- Chen, SY (2012). "Kalman filter for robot vision: a survey". In: *Industrial Electronics, IEEE Transactions on* 59.11, pp. 4409–4420.
- Cook, Diane and Sajal Das (2004). *Smart environments: Technology, protocols and applications*. Vol. 43. John Wiley & Sons.
- Cook, Diane J and Sajal K Das (2007). "How smart are our environments? An updated look at the state of the art". In: *Pervasive and mobile computing* 3.2, pp. 53–73.
- Dagher, Joseph C, Michael W Marcellin, and Mark A Neifeld (2006). "A method for coordinating the distributed transmission of imagery". In: *Image Processing, IEEE Transactions on* 15.7, pp. 1705–1717.
- De Angelis, Alessio et al. (2010). "Indoor positioning by ultrawide band radio aided inertial navigation". In: *Metrology and Measurement Systems* 17.3, pp. 447–460.
- De Santo, Massimo et al. (2004). "Standard uncertainty evaluation in image-based measurements". In: *Measurement* 36.3, pp. 347–358.
- Demiris, George, Brian K Hensel, et al. (2008). "Technologies for an aging society: a systematic review of "smart home" applications". In: *Yearb Med Inform* 3, pp. 33–40.
- Deyle, Travis, Charles C Kemp, and Matthew S Reynolds (2008). "Probabilistic UHF RFID tag pose estimation with multiple antennas and a multipath RF propagation model". In: *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*. IEEE, pp. 1379–1384.
- Di Leo, Giuseppe, Consolatina Liguori, and Alfredo Paolillo (2011). "Covariance propagation for the uncertainty estimation in stereo vision". In: *Instrumentation and Measurement, IEEE Transactions on* 60.5, pp. 1664–1673.
- Diebel, James (2006). "Representing attitude: Euler angles, unit quaternions, and rotation vectors". In: *Matrix* 58, pp. 15–16.
- DLNA. <https://www.dlna.org/>.
- Dünser, Andreas and Eva Hornecker (2007). "An observational study of children interacting with an augmented story book". In: *Technologies for E-Learning and Digital Entertainment*. Springer, pp. 305–315.

- Durrant-Whyte, Hugh (2001). "Introduction to estimation and the Kalman Filter". In: *Australian Centre for Field Robotics* 28.3, pp. 65–94.
- Engel, Jakob, Thomas Schöps, and Daniel Cremers (2014). "LSD-SLAM: Large-scale direct monocular SLAM". In: *Computer Vision—ECCV 2014*. Springer, pp. 834–849.
- Ercan, Ali Özer, A El Gamal, and LJ Guibas (2006). "Camera network node selection for target localization in the presence of occlusions". In:
- Faessler, Matthias et al. (2014). "A monocular pose estimation system based on infrared leds". In: *Robotics and Automation (ICRA), 2014 IEEE International Conference on*. IEEE, pp. 907–913.
- Fan, Qigao et al. (2014). "Integrated navigation fusion strategy of INS/UWB for indoor carrier attitude angle and position synchronous tracking". In: *The Scientific World Journal* 2014.
- Fang, Xi, Dejun Yang, and Guoliang Xue (2012). "Strategizing surveillance for resource-constrained event monitoring". In: *INFOCOM, 2012 Proceedings IEEE*. IEEE, pp. 244–252.
- Feldman, Assaf et al. (2005). "ReachMedia: On-the-move interaction with everyday objects". In: *Wearable Computers, 2005. Proceedings. Ninth IEEE International Symposium on*. IEEE, pp. 52–59.
- Fiala, Mark (2005). "ARTag, a fiducial marker system using digital techniques". In: *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*. Vol. 2. IEEE, pp. 590–596.
- Föckler, Paul et al. (2005). "PhoneGuide: museum guidance supported by on-device object recognition on mobile phones". In: *Proceedings of the 4th international conference on Mobile and ubiquitous multimedia*. ACM, pp. 3–10.
- Foix, Sergi, Guillem Alenya, and Carme Torras (2011). "Lock-in time-of-flight (ToF) cameras: a survey". In: *Sensors Journal, IEEE* 11.9, pp. 1917–1926.
- Franz, Alfred M et al. (2014). "Electromagnetic Tracking in Medicine-A Review of Technology, Validation, and Applications". In: *Medical Imaging, IEEE Transactions on* 33.8, pp. 1702–1725.
- Fuentes-Pacheco, Jorge, José Ruiz-Ascencio, and Juan Manuel Rendón-Mancha (2015). "Visual simultaneous localization and mapping: a survey". In: *Artificial Intelligence Review* 43.1, pp. 55–81.
- Garcia Herrero, J et al. (2001). "Surface movement radar data processing methods for airport surveillance". In: *Aerospace and Electronic Systems, IEEE Transactions on* 37.2, pp. 563–585.
- Gómez, David et al. (2013). "Indoor Augmented Reality Based on Ultrasound Localization Systems". In: *Highlights on Practical Applications of Agents and Multi-Agent Systems*. Springer, pp. 202–212.
- Grewal, Mohinder S (2011). *Kalman filtering*. Springer.
- Grillo, Antonio et al. (2010). "High capacity colored two dimensional codes". In: *Computer Science and Information Technology (IMCSIT), Proceedings of the 2010 International Multiconference on*. IEEE, pp. 709–716.
- Gustafsson, Fredrik et al. (2002). "Particle filters for positioning, navigation, and tracking". In: *Signal Processing, IEEE Transactions on* 50.2, pp. 425–437.

- Hart, Jane K and Kirk Martinez (2006). "Environmental Sensor Networks: A revolution in the earth system science?" In: *Earth-Science Reviews* 78.3, pp. 177–191.
- Hartley, Richard and Andrew Zisserman (2003). *Multiple view geometry in computer vision*. Cambridge university press.
- Hartley, Richard I and Peter Sturm (1997). "Triangulation". In: *Computer vision and image understanding* 68.2, pp. 146–157.
- Hees, Vincent T van et al. (2013). "Separating movement and gravity components in an acceleration signal and implications for the assessment of human daily physical activity". In: *PLoS One* 8.4, e61691.
- Helal, Sumi et al. (2005). "The gator tech smart house: A programmable pervasive space". In: *Computer* 38.3, pp. 50–60.
- Henry, Peter et al. (2012). "RGB-D mapping: Using Kinect-style depth cameras for dense 3D modeling of indoor environments". In: *The International Journal of Robotics Research* 31.5, pp. 647–663.
- Höllerer, Tobias and Steve Feiner (2004). "Mobile augmented reality". In: *Telegeoinformatics: Location-Based Computing and Services*. Taylor and Francis Books Ltd., London, UK 21.
- HTC Vive. <https://www.htcvive.com/>.
- Huang, Albert S et al. (2011). "Visual odometry and mapping for autonomous flight using an RGB-D camera". In: *International Symposium on Robotics Research (ISRR)*, pp. 1–16.
- Jamal, Syed Zahid (2012). "Tightly coupled GPS/INS airborne navigation system". In: *Aerospace and Electronic Systems Magazine, IEEE* 27.4, pp. 39–42.
- Janabi-Sharifi, Farrokh and Mohammed Marey (2010). "A kalman-filter-based method for pose estimation in visual servoing". In: *Robotics, IEEE Transactions on* 26.5, pp. 939–947.
- Jiang, Bolan, Ulrich Neumann, and Suyu You (2004). "A robust hybrid tracking system for outdoor augmented reality". In: *null. IEEE*, p. 3.
- Joshi, Kinjal A and Darshak G Thakore (2012). "A survey on moving object detection and tracking in video surveillance system". In: *IJSCE, ISSN*, pp. 2231–2307.
- Julier, Simon J and Jeffrey K Uhlmann (2004). "Unscented filtering and nonlinear estimation". In: *Proceedings of the IEEE* 92.3, pp. 401–422.
- Kalman, Rudolph Emil (1960). "A new approach to linear filtering and prediction problems". In: *Journal of Fluids Engineering* 82.1, pp. 35–45.
- Kan, Tai-Wei, Chin-Hung Teng, and Wen-Shou Chou (2009). "Applying QR code in augmented reality applications". In: *Proceedings of the 8th International Conference on Virtual Reality Continuum and its Applications in Industry*. ACM, pp. 253–257.
- Kannisto, Juha et al. (2005). "Software and hardware prototypes of the IEEE 1588 precision time protocol on wireless LAN". In: *Local and Metropolitan Area Networks, 2005. LANMAN 2005. The 14th IEEE Workshop on*. IEEE, 6–pp.
- Karasalo, Maja and Xiaoming Hu (2011). "An optimization approach to adaptive Kalman filtering". In: *Automatica* 47.8, pp. 1785–1793.
- Kelly, Jonathan and Gaurav S Sukhatme (2011). "Visual-inertial sensor fusion: Localization, mapping and sensor-to-sensor self-calibration". In: *The International Journal of Robotics Research* 30.1, pp. 56–79.

- Khalfin, Igor and Amir Rubin (2013). *Electromagnetic Tracker (AC) with Extended Range and Distortion Compensation Capabilities Employing Multiple Transmitters*. US Patent App. 13/796,210.
- Kim, Kyungnam and Larry S Davis (2006). "Multi-camera tracking and segmentation of occluded people on ground plane using search-guided particle filtering". In: *Computer Vision—ECCV 2006*. Springer, pp. 98–109.
- Kim, SoonWhan and Jin-Shig Kang (2014). "A New Outdoor Object Tracking Approach in Video Surveillance". In: *Advanced Intelligent Systems*. Springer, pp. 167–177.
- Kinect, Microsoft. *Kinect*. <https://www.microsoft.com/en-us/kinectforwindows/>.
- Klein, Georg and Tom Drummond (2004). "Sensor fusion and occlusion refinement for tablet-based AR". In: *Mixed and Augmented Reality, 2004. ISMAR 2004. Third IEEE and ACM International Symposium on*. IEEE, pp. 38–47.
- Kodaka, Kenri et al. (2008). "Pose estimation of a mobile robot on a lattice of RFID tags". In: *Intelligent Robots and Systems, 2008. IROS 2008. IEEE/RSJ International Conference on*. IEEE, pp. 1385–1390.
- Kohler, Markus et al. (1997). *Using the Kalman filter to track human interactive motion: modelling and initialization of the Kalman filter for translational motion*. Citeseer.
- Kovavisaruch, La-or et al. (2012). "Evaluating and collecting museum visitor behavior via RFID". In: *Technology Management for Emerging Technologies (PICMET), 2012 Proceedings of PICMET'12: IEEE*, pp. 1099–1101.
- Kwapisz, Jennifer R, Gary M Weiss, and Samuel A Moore (2011). "Activity recognition using cell phone accelerometers". In: *ACM SigKDD Explorations Newsletter* 12.2, pp. 74–82.
- Kwon, Ohyun, Jeongho Shin, and Joonki Paik (2006). "Edge based adaptive Kalman filtering for real-time video stabilization". In: *Proc. Int. Conf. Consum. Electron*, pp. 75–76.
- Kyriakoulis, Nikolaos and Antonios Gasteratos (2010). "Color-based monocular visuoinertial 3-D pose estimation of a volant robot". In: *Instrumentation and Measurement, IEEE Transactions on* 59.10, pp. 2706–2715.
- Leap Motion. <https://www.leapmotion.com/>.
- Lee, Young-Sook and Wan-Young Chung (2012). "Visual sensor based abnormal event detection with moving shadow removal in home healthcare applications". In: *Sensors* 12.1, pp. 573–584.
- Lefferts, Ern J, F Landis Markley, and Malcolm D Shuster (1982). "Kalman filtering for spacecraft attitude estimation". In: *Journal of Guidance, Control, and Dynamics* 5.5, pp. 417–429.
- Li, J. et al. (2016). "Camera pose estimation by vision-inertial sensor fusion: an application to augmented reality books". In: *IS&T International Symposium on Electronic Imaging 2016*. Accepted. San Francisco, US.
- Li, Juan et al. (2015a). "A combined vision-inertial fusion approach for 6-DoF object pose estimation". In: *Seventh International Conference on Machine Vision (ICMV 2014)*. International Society for Optics and Photonics, pp. 944518–944518.
- Li, Juan et al. (2015b). "A hybrid pose tracking approach for handheld augmented reality". In: *Proceedings of the 9th International Conference on Distributed Smart Camera*. ACM, pp. 7–12.
- Li, Juan et al. (2015c). "A new 360-degree immersive game controller". In: *Proceedings of the 9th International Conference on Distributed Smart Camera*. ACM, pp. 201–202.

- Li, Juan et al. (2017a). "A Novel Laser-vision Combined Approach to Indoor Robot Pose Estimation". In: *International Journal of Computer Theory and Engineering* 9.2, pp. 103–108.
- Li, Juan et al. (2017b). "A novel system for object pose estimation using fused vision and inertial data". In: *Information Fusion* 33, pp. 15–28.
- Li, X Rong and Vesselin P Jilkov (2005). "Survey of maneuvering target tracking. Part V. Multiple-model methods". In: *Aerospace and Electronic Systems, IEEE Transactions on* 41.4, pp. 1255–1321.
- Li, Xin et al. (2010). "A multiple object tracking method using Kalman filter". In: *Information and Automation (ICIA), 2010 IEEE International Conference on*. IEEE, pp. 1862–1866.
- Li, Yiming and Bir Bhanu (2011a). "A Comparison of Techniques for Camera Selection and Hand-Off in a Video Network". In: *Distributed Video Sensor Networks*. Springer, pp. 69–83.
- Li, Yiming and Bir Bhanu (2011b). "Utility-based camera assignment in a video network: A game theoretic framework". In: *Sensors Journal, IEEE* 11.3, pp. 676–687.
- Ligorio, Gabriele and Angelo Maria Sabatini (2013). "Extended Kalman filter-based methods for pose estimation using visual, inertial and magnetic sensors: comparative analysis and performance evaluation". In: *Sensors* 13.2, pp. 1919–1941.
- Lippiello, Vincenzo, Bruno Siciliano, and Luigi Villani (2002). "Objects motion estimation via BSP tree modeling and Kalman filtering of stereo images". In: *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*. Vol. 3. IEEE, pp. 2968–2973.
- Lippiello, Vincenzo, Bruno Siciliano, and Luigi Villani (2007). "Adaptive extended Kalman filtering for visual motion estimation of 3D objects". In: *Control Engineering Practice* 15.1, pp. 123–134.
- Liu, Hui et al. (2007). "Survey of wireless indoor positioning techniques and systems". In: *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on* 37.6, pp. 1067–1080.
- Liu, Liang, Xi Zhang, and Huadong Ma (2009). "Dynamic node collaboration for mobile target tracking in wireless camera sensor networks". In: *INFOCOM 2009, IEEE*. IEEE, pp. 1188–1196.
- Liu, Xiaolan (2014). "A survey on wireless camera sensor networks". In: *Frontier and Future Development of Information Technology in Medicine and Education*. Springer, pp. 1085–1094.
- Lowe, David G (2004). "Distinctive image features from scale-invariant keypoints". In: *International journal of computer vision* 60.2, pp. 91–110.
- Mahalanobis, Prasanta Chandra (1936). "On the generalized distance in statistics". In: *Proceedings of the National Institute of Sciences (Calcutta)* 2, pp. 49–55.
- Marshall, Catherine C (2005). "Reading and interactivity in the digital library: Creating an experience that transcends paper". In: *Proceedings of CLIR/Kanazawa Institute of Technology Roundtable*. Vol. 5. 4. Citeseer, pp. 1–20.
- Martin, Henar et al. (2014). "Simplified Pedestrian Tracking Filters with Positioning and Foot-Mounted Inertial Sensors". In: *International Journal of Distributed Sensor Networks* 2014.
- Martínez, Carol et al. (2011). "On-board and ground visual pose estimation techniques for UAV control". In: *Journal of Intelligent & Robotic Systems* 61.1-4, pp. 301–320.

- Mazor, Efim et al. (1998). "Interacting multiple model methods in target tracking: a survey". In: *Aerospace and Electronic Systems, IEEE Transactions on* 34.1, pp. 103–123.
- Miyashita, Tsutomu et al. (2008). "An augmented reality museum guide". In: *Proceedings of the 7th IEEE/ACM International Symposium on Mixed and Augmented Reality*. IEEE Computer Society, pp. 103–106.
- Mohring, Mathias, Christian Lessig, and Oliver Bimber (2004). "Video see-through ar on consumer cell-phones". In: *Proceedings of the 3rd IEEE/ACM International Symposium on Mixed and Augmented Reality*. IEEE Computer Society, pp. 252–253.
- Monari, Eduardo and Kristian Kroschel (2010). "Dynamic sensor selection for single target tracking in large video surveillance networks". In: *Advanced Video and Signal Based Surveillance (AVSS), 2010 Seventh IEEE International Conference on*. IEEE, pp. 539–546.
- Mozer, Michael (2004). "Lessons from an adaptive house". PhD thesis. Architectural Engineering.
- Naimark, Leonid and Eric Foxlin (2002). "Circular data matrix fiducial system and robust image processing for a wearable vision-inertial self-tracker". In: *Proceedings of the 1st International Symposium on Mixed and Augmented Reality*. IEEE Computer Society, p. 27.
- Newcombe, Richard A et al. (2011). "KinectFusion: Real-time dense surface mapping and tracking". In: *Mixed and augmented reality (ISMAR), 2011 10th IEEE international symposium on*. IEEE, pp. 127–136.
- Nintendo Wii Remote. <https://www.wii.com/>.
- Nocedal, Jorge and Stephen J Wright (2006). "Least-Squares Problems". In: *Numerical Optimization*, pp. 245–269.
- Oculus Rift. <https://www.oculus.com/>.
- Oh, Hyondong et al. (2011). "Indoor UAV control using multi-camera visual feedback". In: *Journal of Intelligent & Robotic Systems* 61.1-4, pp. 57–84.
- Olson, Edwin (2011). "AprilTag: A robust and flexible visual fiducial system". In: *Robotics and Automation (ICRA), 2011 IEEE International Conference on*. IEEE, pp. 3400–3407.
- OpenCV. <https://www.opencv.org>.
- Optitrack. <https://www.optitrack.com/>.
- Pahalawatta, Peshala V, Thrasyvoulos N Pappas, and Aggelos K Katsaggelos (2004). "Optimal sensor selection for video-based target tracking in a wireless sensor network". In: *Image Processing, 2004. ICIP'04. 2004 International Conference on*. Vol. 5. IEEE, pp. 3073–3076.
- Park, Johnny, Priya C Bhat, and Avinash C Kak (2006). "A look-up table based approach for solving the camera selection problem in large camera networks". In: *Proceedings of the International Workshop on Distributed Smart Cameras (DCS'06), Boulder, CO, Oct.* Vol. 31, pp. 72–76.
- Park, Minha and Yang Gao (2008). "Error and performance analysis of MEMS-based inertial sensors with a low-cost GPS receiver". In: *Sensors* 8.4, pp. 2240–2261.
- Pedley, Mark (2013). "Tilt sensing using a three-axis accelerometer". In: *Freescale Semiconductor Application Note*.

- Pertile, M et al. (2014). "Position and orientation measurement of a fast moving multibody system in ground tests". In: *Metrology for Aerospace (MetroAeroSpace), 2014 IEEE*. IEEE, pp. 445–449.
- Peterson, Diana and Cynthia Robertson (2013). "Utilizing virtual reality in teaching and training: Progress toward virtual surgical simulations". In: *INTED2013 proceedings*, pp. 4–5.
- Piccardi, Massimo (2004). "Background subtraction techniques: a review". In: *Systems, man and cybernetics, 2004 IEEE international conference on*. Vol. 4. IEEE, pp. 3099–3104.
- Pintaric, Thomas and Hannes Kaufmann (2007). "Affordable infrared-optical pose-tracking for virtual and augmented reality". In: *Proceedings of Trends and Issues in Tracking for Virtual Environments Workshop, IEEE VR*, pp. 44–51.
- Pitre, Ryan R, Vesselin P Jilkov, and X Rong Li (2005). "A comparative study of multiple-model algorithms for maneuvering target tracking". In: *Defense and Security*. International Society for Optics and Photonics, pp. 549–560.
- Prochazka, David and Tomas Koubek (2011). "Augmented reality implementation methods in mainstream applications". In: *arXiv preprint arXiv:1106.5569*.
- Qadri, Muhammad Tahir and Muhammad Asif (2009). "Automatic number plate recognition system for vehicle identification using optical character recognition". In: *Education Technology and Computer, 2009. ICETC'09. International Conference on*. IEEE, pp. 335–338.
- Rampinelli, Mariana et al. (2014). "An intelligent space for mobile robot localization using a multi-camera system". In: *Sensors* 14.8, pp. 15039–15064.
- Rautaray, Siddharth S and Anupam Agrawal (2015). "Vision based hand gesture recognition for human computer interaction: a survey". In: *Artificial Intelligence Review* 43.1, pp. 1–54.
- Rolland, Jannick P, Larry Davis, and Yohan Baillot (2001). "A survey of tracking technology for virtual environments". In: *Fundamentals of wearable computers and augmented reality* 1, pp. 67–112.
- Ronzhin, Alexander, Maria Prischepa, and Alexey Karpov (2010). "A video monitoring model with a distributed camera system for the smart space". In: *Smart Spaces and Next Generation Wired/Wireless Networking*. Springer, pp. 102–110.
- Rowaihy, Hosam et al. (2007). "A survey of sensor selection schemes in wireless sensor networks". In: *Defense and Security Symposium*. International Society for Optics and Photonics, 65621A–65621A.
- Sabatini, Angelo M (2006). "Quaternion-based extended Kalman filter for determining orientation by inertial and magnetic sensing". In: *Biomedical Engineering, IEEE Transactions on* 53.7, pp. 1346–1356.
- Sahayadhas, Arun, Kenneth Sundaraj, and Murugappan Murugappan (2012). "Detecting driver drowsiness based on sensors: a review". In: *Sensors* 12.12, pp. 16937–16953.
- Satoh, Kiyohide, Shinji Uchiyama, and Hiroyuki Yamamoto (2004). "A head tracking method using bird's-eye view camera and gyroscope". In: *Proceedings of the 3rd IEEE/ACM International Symposium on Mixed and Augmented Reality*. IEEE Computer Society, pp. 202–211.
- Schall, Gerhard et al. (2009). "Global pose estimation using multi-sensor fusion for outdoor augmented reality". In: *Mixed and Augmented Reality, 2009. ISMAR 2009. 8th IEEE International Symposium on*. IEEE, pp. 153–162.

- Schmalstieg, Dieter and Daniel Wagner (2009). "Mobile phones as a platform for augmented reality". In: *connections* 1, p. 3.
- Shademan, Azad and Farrokh Janabi-Sharifi (2005). "Sensitivity analysis of EKF and iterated EKF pose estimation for position-based visual servoing". In: *Control Applications, 2005. CCA 2005. Proceedings of 2005 IEEE Conference on*. IEEE, pp. 755–760.
- Shen, Edward and Richard Hornsey (2011). "Local image quality metric for a distributed smart camera network with overlapping FOVs". In: *Distributed Smart Cameras (ICDSC), 2011 Fifth ACM/IEEE International Conference on*. IEEE, pp. 1–6.
- Sinan, Gezici, Tian Zhi, B Giannakis Georgios, et al. (2005). "Localization via ultra-wideband radios". In: *IEEE, Signal Processing Magazine* 35.2, pp. 131–135.
- Sony PlayStation. <https://www.playstation.com/>.
- Soro, Stanislava and Wendi Heinzelman (2007). "Camera selection in visual sensor networks". In: *Advanced Video and Signal Based Surveillance, 2007. AVSS 2007. IEEE Conference on*. IEEE, pp. 81–86.
- Soro, Stanislava and Wendi Heinzelman (2009). "A survey of visual sensor networks". In: *Advances in Multimedia* 2009.
- Soto, Cristian, Bi Song, and Amit K Roy-Chowdhury (2009). "Distributed multi-target tracking in a self-configuring camera network". In: *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. IEEE, pp. 1486–1493.
- Sun, Guolin et al. (2005). "Signal processing techniques in network-aided positioning: a survey of state-of-the-art positioning designs". In: *Signal Processing Magazine, IEEE* 22.4, pp. 12–23.
- Tarrío, Paula, Ana M Bernardos, and José R Casar (2011). "Weighted least squares techniques for improved received signal strength based localization". In: *Sensors* 11.9, pp. 8569–8592.
- Tessens, Linda et al. (2008). "Principal view determination for camera selection in distributed smart camera networks". In: *Distributed Smart Cameras, 2008. ICDSC 2008. Second ACM/IEEE International Conference on*. IEEE, pp. 1–10.
- Tessens, Linda et al. (2014). "Camera selection for tracking in distributed smart camera networks". In: *ACM Transactions on Sensor Networks (TOSN)* 10.2, p. 23.
- Unity. <https://www.unity3d.com/>.
- Ventura, Jordi et al. (2014). "Global localization from monocular SLAM on a mobile phone". In: *Visualization and Computer Graphics, IEEE Transactions on* 20.4, pp. 531–539.
- Vicon Motion Capture System. <http://www.vicon.com/>.
- Wagner, Daniel et al. (2010). "Real-time detection and tracking for augmented reality on mobile phones". In: *Visualization and Computer Graphics, IEEE Transactions on* 16.3, pp. 355–368.
- Wang, Jiang and William J Wilson (1992). "3D relative position and orientation estimation using Kalman filter for robot control". In: *Robotics and Automation, 1992. Proceedings., 1992 IEEE International Conference on*. IEEE, pp. 2638–2645.
- Wang, Xian et al. (2015). "A gesture-based method for natural interaction in smart spaces". In: *Journal of Ambient Intelligence and Smart Environments* 7.4, pp. 535–562.
- Wang, Xiaogang (2013). "Intelligent multi-camera video surveillance: A review". In: *Pattern recognition letters* 34.1, pp. 3–19.

- Wang, Yong and Dianhong Wang (2013). "Energy-efficient node selection for target tracking in wireless sensor networks". In: *International Journal of Distributed Sensor Networks* 2013.
- Wang, Yong, Dianhong Wang, and Wu Fang (2014). "Automatic node selection and target tracking in wireless camera sensor networks". In: *Computers & Electrical Engineering* 40.2, pp. 484–493.
- Weiser, Mark (1991). "The computer for the 21st century". In: *Scientific american* 265.3, pp. 94–104.
- Wells, Curt (2013). *The Kalman filter in finance*. Vol. 32. Springer Science & Business Media.
- Wilson, Emmanuel et al. (2007). "A hardware and software protocol for the evaluation of electromagnetic tracker accuracy in the clinical environment: a multi-center study". In: *Medical Imaging*. International Society for Optics and Photonics, 65092T–65092T.
- Wu, Hsin-Kai et al. (2013). "Current status, opportunities and challenges of augmented reality in education". In: *Computers & Education* 62, pp. 41–49.
- Xiao, Wendong, Wei Ni, and Yue Khing Toh (2011). "Integrated Wi-Fi fingerprinting and inertial sensing for indoor positioning". In: *Indoor Positioning and Indoor Navigation (IPIN), 2011 International Conference on*. IEEE, pp. 1–6.
- Yang, Hyun S et al. (2008). "Hybrid visual tracking for augmented books". In: *Entertainment Computing-ICEC 2008*. Springer, pp. 161–166.
- Yang, Zhenyu and Klara Nahrstedt (2005). "A bandwidth management framework for wireless camera array". In: *Proceedings of the international workshop on Network and operating systems support for digital audio and video*. ACM, pp. 147–152.
- Yilmaz, Alper, Omar Javed, and Mubarak Shah (2006). "Object tracking: A survey". In: *Acm computing surveys (CSUR)* 38.4, p. 13.
- You, Suyu, Ulrich Neumann, and Ronald Azuma (1999). "Hybrid inertial and vision tracking for augmented reality registration". In: *Virtual Reality, 1999. Proceedings., IEEE*. IEEE, pp. 260–267.
- Yu, Chao and Gaurav Sharma (2010). "Camera scheduling and energy allocation for lifetime maximization in user-centric visual sensor networks". In: *Image Processing, IEEE Transactions on* 19.8, pp. 2042–2055.
- Yu, Ying Kin, Kin Hong Wong, and Michael Ming Yuen Chang (2005). "Recursive three-dimensional model reconstruction based on Kalman filtering". In: *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on* 35.3, pp. 587–592.
- Yun, Xiaoping et al. (2003). "An improved quaternion-based Kalman filter for real-time tracking of rigid body orientation". In: *Intelligent Robots and Systems, 2003.(IROS 2003). Proceedings. 2003 IEEE/RSJ International Conference on*. Vol. 2. IEEE, pp. 1074–1079.
- Yut, Libei, Qing Fei, and Qingbo Geng (2013). "Combining Zigbee and inertial sensors for quadrotor UAV indoor localization". In: *2013 10th IEEE International Conference on Control and Automation (ICCA)*. IEEE, pp. 1912–1916.
- Zhang, Xiang, Stephan Frönz, and Nassir Navab (2002). "Visual marker detection and decoding in AR systems: A comparative study". In: *Proceedings of the 1st International Symposium on Mixed and Augmented Reality*. IEEE Computer Society, p. 97.

- Zhang, Zhengyou (1999). "Flexible camera calibration by viewing a plane from unknown orientations". In: *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*. Vol. 1. IEEE, pp. 666–673.
- Zhou, Feng, Henry Been-Lirn Duh, and Mark Billinghurst (2008). "Trends in augmented reality tracking, interaction and display: A review of ten years of ISMAR". In: *Proceedings of the 7th IEEE/ACM International Symposium on Mixed and Augmented Reality*. IEEE Computer Society, pp. 193–202.