REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique
Université Mohamed Khider – BISKRA
Faculté des Sciences Exactes, des Sciences de la Nature et de la Vie
Département d'informatique

N° d'ordre : /M2/2020

Mémoire

présenté pour obtenir le diplôme de master académique en

Informatique

Parcours : **Images et vie artificielle**

Handling occlusions based on 3D reconstruction in augmented reality

---

Par :

MESSERHI DJALLEL

Soutenu le septembre 2020, devant le jury composé de :

Nom et prénom	Grade	Président
Babahenini Mohamed Chaouki	Professeur	Rapporteur
Nom et prénom	Grade	Examinateur

Dedication

I dedicate this work

To my dear parents

To my sister

All of my family

To my friends and my colleagues …

Acknowledgement

Before all, my deep and sincere praise to Allah the almighty for providing me with the strength and patience to accomplish this research work.

My thanks go to my supervisor Dr. Babahenini Mohamed Chaouki for his valuable guidance.

Special thanks would go to the members of the jury namely, ~~name 1~~, ~~name 2~~ for their efforts to evaluate my research work. I also express my gratitude to my classmate Bekiri Roumaissa for her tips through this work.

Thanks also go to all my friends with whom I shared unforgettable moments during my studies.

Finally, all my gratitude, my gratitude and my very warm thanks to all those who have contributed from near or far and all the teachers of the computer science department of Biskra, in particular Dr. Babahenini Mohamed Chaouki, Dr. Zerari Abdemoumene,

Dr. Djerou Leila, Dr. Rahmani Salima, to my training of graduate studies and of post-graduation.

Abstract

The correct relationships between real and virtual objects are of utmost importance to realistic augmented reality system, in which the occlusion handling method should be able to estimate the spatial relationships between real and virtual objects, as well as handle the mutual occlusion automatically in real-time.

In this work, we propose two occlusion handling methods based on 3D reconstruction: Photogrammetry for 3D reconstruction method and Real-time 3D reconstruction method.

More specifically, we can judge and handle the mutual occlusion without human interactivity in real-time and experimental results prove its effectiveness.

Keywords: Augmented Reality, handling occlusions, 3D reconstruction, Photogrammetry, Real-time 3D reconstruction.

ملخص

العلاقات الصحيحة بين الكائنات الحقيقية والافتراضية ذات أهمية قصوى لنظام الواقع المعزز الواقعي ، حيث يجب أن تكون طريقة معالجة الانسداد قادرة على تقدير العلاقات المكانية بين الكائنات الحقيقية والافتراضية ، وكذلك التعامل مع الانسداد المتبادل تلقائيًا في الوقت الفعلي.

في هذا العمل ، نقترح طريقتين لمعالجة الانسداد استنادًا إلى إعادة الإعمار ثلاثية الأبعاد: القياس التصويري لطريقة إعادة البناء ثلاثية الأبعاد وطريقة إعادة البناء ثلاثية الأبعاد في الوقت الفعلي.

وبشكل أكثر تحديدًا ، يمكننا الحكم على الانسداد المتبادل والتعامل معه دون التفاعل البشري في الوقت الفعلي والنتائج التجريبية تثبت فعاليتها.

الكلمات المفتاحية : الواقع المعزز ، التعامل مع الانسدادات ، إعادة البناء ثلاثية الأبعاد ، القياس التصويري ، إعادة الإعمار ثلاثية الأبعاد في الوقت الفعلي.

Contents

List of Figures

List of Tables

ABBREVIATIONS

AR	Augmented Reality.
VR	Virtual Reality.
AV	Augmented Virtuality.
VE	Virtual Environment.
RE	Real Environment.
MR	Mixed Reality.
XR	Extended Reality.
2D	2 Dimensional.
3D	3 Dimensional.
SLAM	Simultaneous Localization and Mapping.
SFM	Structure From Motion.
HMD	Head Mounted Displays.
SAR	Spatial Augmented Reality.
GPS	Global Positioning System.
CNR	National Council of Research.
ITIA	Institute of Industrial Technologies and Automation.
DSLR	Digital Single Lens Reflex.
POI	Point Of Interest.
CAD	Computer-Aided Design.
CAM	Computer-Aided Manufacturing.
JPEG	Joint Photographic Experts Group.
CPU	Central Processing Unit.
GPU	Graphics Processing Unit.
JSON	JavaScript Object Notation.
SIFT	Scale-Invariant Feature Transform.
RANSAC	RANdom SAmple Consensus.
PnP	Perspective-n-Point algorithm.

SGM	Semi-Global Matching.
ZNCC	Zero Mean Normalized Cross-Correlation.
ICP	Iterative Closest Point.
LiDAR	Light Detection and Ranging.
IDE	Integrated development environment.
API	Application Program Interface.
SDK	Software Development Kit.
UAV	Unmanned Aerial Vehicles.
MP	Mega Pixels.
CMOS	Complementary Metal Oxide Semiconductor.
OS	Operating System.
GUI	Graphical User Interface.
CLI	Command Line.
MB	Megabytes.
fps	frames per second.
mm	millimeters.

General Introduction

The technological developments of the last twenty years have allowed digital systems to invade our daily lives. There is more to demonstrate that digital occupies a place of choice in the world today. Among the major components of digital systems, great importance is given to the image. Augmented reality and Virtual reality is the subject of very active research at present. Augmented reality is a very vast field that has known, and still knows, an important development for a few decades.

Augmented reality (AR) systems aim at adding virtual objects to the real scene to make the virtual objects merge with the existing world seamlessly in such a manner as to appear part of the viewed 3D scene. In contrast with virtual reality, users can see virtual objects and the real world simultaneously in the augmented reality system. Augmented reality has come a long way in recent years, it has applications everywhere. The applications of augmented reality are extensive: computer-aided surgery, entertainment, education, tourism industry, military exercises, and product design.

Researchers have paid more and more attention to the occlusion problem in augmented reality. The problem of occlusion occurs when the real objects are in front of the virtual objects in the scene. Users will have the misconception that the real object is further from the viewpoint than the virtual objects.

In this master project, we focus on methods handling occlusions in augmented reality in order to make the user feel that the real objects and the virtual objects are mixing in the scene. To do this, we propose two methods based: Photogrammetry for 3D reconstruction method and Real-time 3D reconstruction method.

Our manuscript is organized in 4 chapters with a general introduction and a conclusion, and is organized as follows: In the first chapter is devoted to the concepts of augmented reality so that readers are familiar with the field of augmented reality and its technologies and the Applications. The second chapter presents the theory part of methods handling occlusions. The third chapter deals with the design of our solution in the two methods. The fourth chapter deals with the implementation of two methods and the results obtained.

Chapter 1

1. Augmented reality

1.1. Introduction

We define Augmented Reality (AR) as a real-time direct or indirect view of a physical real-world environment that has been enhanced / augmented by adding virtual computer-generated information to it [1] .AR is both interactive and registered in 3D as well as combines real and virtual objects. Milgram's Reality-Virtuality Continuum is defined by Paul Milgram and Fumio Kishino as a continuum that spans between the real environment and the virtual environment comprise Augmented Reality and Augmented Virtuality (AV) in between, where AR is closer to the real world and AV is closer to a pure virtual environment, as seen in Fig 1.1 [2].

This chapter provides the basic notions of augmented reality. After the general definition of augmented reality, of Computer vision methods in AR and of all its technologies and Applications, then we illustrate the challenges of occlusions in augmented reality.

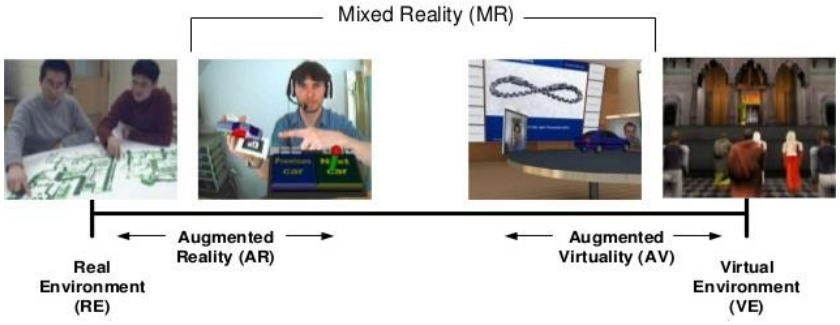


Figure 1.1: Milgram's Reality-Virtuality Continuum [1].

1.2. Augmented reality technologies

1.2.1. Computer vision methods in AR

Computer vision renders 3D virtual objects from the same viewpoint from which the images of the real scene are being taken by tracking cameras. Augmented reality image registration uses different method of computer vision mostly related to video tracking. These methods usually consist of two stages: tracking and reconstructing/recognizing.

First, fiducial markers, optical images, or interest points are detected in the camera images. Tracking can make use of feature detection, edge detection, or other image processing methods to interpret the camera images. In computer vision, most of the available tracking techniques can be separated in two classes: feature-based and model-based [3].

Feature-based methods consist of discovering the connection between 2D image features and their 3D world frame coordinates [4].

Model-based methods make use of model of the tracked objects' features such as CAD (computer-aided design) models or 2D templates of the item based on distinguishable features [3].

Once a connection is made between the 2D image and 3D world frame, it is possible to find the camera pose by projecting the 3D coordinates of the feature into the observed 2D image coordinates and by minimizing the distance to their corresponding 2D features. The constraints for camera pose estimation are most often determined using point features. The reconstructing/ recognizing stage uses the data obtained from the first stage to reconstruct a real world coordinate system.

Assuming a calibrated camera and a perspective projection model, if a point has coordinates $(x, y, z)^T$ in the coordinate frame of the camera, its projection onto the image plane is $(x/z, y/z, 1)^T$.

In point constraints, we have two principal coordinate systems, as illustrated in Fig 1.2, the world coordinate system W and the 2D image coordinate system. Let $\boldsymbol{p_i(x_i, y_i, z_i)^T}$, where $\boldsymbol{i = 1, \dots, n}$, with **n≥3**, be a set of 3D non-collinear reference points in the world

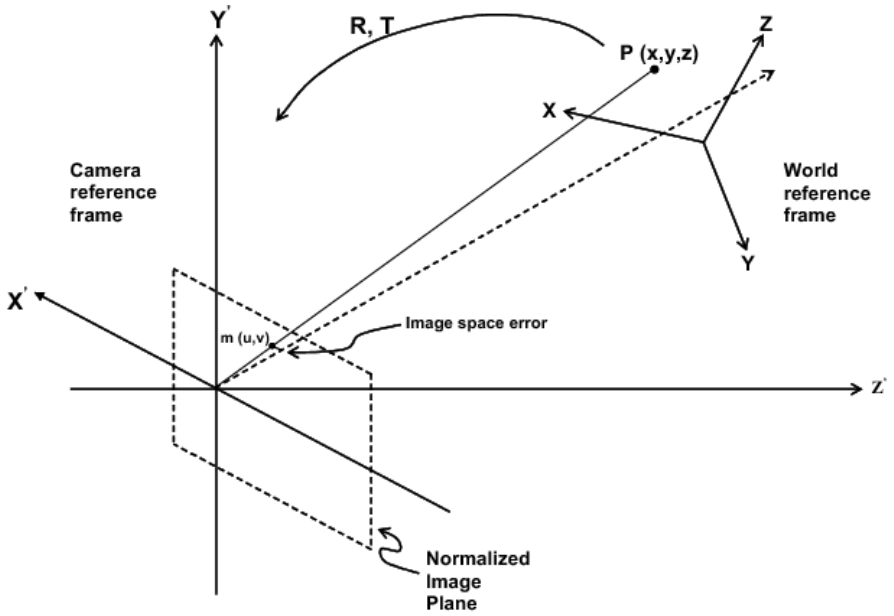


Figure 1.2: Point constraints for the camera pose problem adapted from [5].

frame coordinate and $\boldsymbol{q_i(x_i', y_i', z_i')^T}$ be the corresponding camera-space coordinates, $\boldsymbol{p_i}$ and $\boldsymbol{q_i}$ are related by the following transformation:

$$\boldsymbol{q_i = Rp_i + T}$$

$$\textbf{where R} = \begin{pmatrix} \boldsymbol{r_1^T} \\ \boldsymbol{r_2^T} \\ \boldsymbol{r_3^T} \end{pmatrix} \boldsymbol{and\ T} = \begin{pmatrix} \boldsymbol{t_x} \\ \boldsymbol{t_y} \\ \boldsymbol{t_z} \end{pmatrix}$$

are a rotation matrix and a translation vector, respectively.

Let the image point $\boldsymbol{h_i(u_i, v_i, 1)^T}$ be the projection of $\boldsymbol{p_i}$ on the normalized image plane. The collinearity equation establishing the relationship between $\boldsymbol{h_i}$ and $\boldsymbol{P_i}$ using the camera pinhole is given by:

$$h_i = \frac{1}{r_3^T p_i + t_z}(Rp_i + T)$$

The image space error gives a relationship between 3D reference points, their corresponding 2D extracted image points, and the camera pose parameters, and corresponds to the point constraints [1]. The image space error is given as follow:

$$E_i^p = \sqrt{\left(\hat{u}_i - \frac{r_1^T p_i + t_x}{r_3^T p_i + t_z}\right)^2 + \left(\hat{v}_i - \frac{r_2^T p_i + t_y}{r_3^T p_i + t_z}\right)^2}$$

where $\hat{\mathbf{m}}_i(\hat{u}_i, \hat{v}_i, 1)^T$ are the observed image points.

Some methods assume the presence of fiducial markers in the environment or object with known 3D geometry, and make use of those data **[6]**. however, the device will have to be stationary and its position known. If the entire scene is not known beforehand, Simultaneous Localization and Mapping (SLAM) technique is used for mapping fiducial markers or 3D models relative positions. In the case when no assumptions about the 3Dgeometry of the scene can be made, Structure from Motion (SFM) method is used. SFM method can be divided into two parts: feature point tracking and camera parameter estimation.

Tracking methods in AR depend mostly on the type of environment the AR device will be introduced to as well as the type of AR system. The environment might be indoor, outdoor or a combination of both. In the same way, the system might be mobile or static (have a fixed-position). For example, if the AR device is a fixed-position device for an outdoor real environment **[6]**, the developers can use mechanical tracking since the movements to be tracked will all be mechanical, as the position of the device is known. This type of environment and system makes tracking of the environment for augmenting the surroundings easier. On the other hand, if the AR device is mobile and designed for an outdoor environment, tracking becomes much harder and different techniques offer some advantages and disadvantages. For example, built a pedestrian detection system for automotive collision avoidance using AR **[7]**.Their system is mobile and outdoor. For a camera moving in an unknown environment, the problem for computer vision is to reconstruct both the motion of the camera and the structure of the scene using the image and additional sensor data sequences. In this case, since no assumption about the 3D geometry of the scene can be made, SfM method is used for reconstructing the scene.

Developers also have the choice to make use of existing Vuforia, this is an augmented reality software development kit (SDK) for mobile devices that enables the creation of augmented reality applications. [8] It uses computer vision technology to recognize and track planar images and 3D objects in real time. This image registration capability enables developers to position and orient virtual objects, such as 3D models and other media, in relation to real world objects when they are viewed through the camera of a mobile device. The virtual object then tracks the position and orientation of the image in real-time so that the viewer's perspective on the object corresponds with the perspective on the target. It thus appears that the virtual object is a part of the real-world scene.

The Vuforia SDK supports a variety of 2D and 3D target types including ‘markerless’ Image Targets, 3D Model Target, and a form of addressable Fiducial Marker, known as a VuMark.

Additional features of the SDK include 6 degrees of freedom device localization in space, localized Occlusion Detection using 'Virtual Buttons', runtime image target selection, and the ability to create and reconfigure target sets programmatically at runtime [9].

Vuforia provides Application Programming Interfaces (API) in C++, Java, Objective-C++, and the .NET languages through an extension to the Unity game engine. **[10]** In this way, the SDK supports both native development for iOS, Android, and UWP while it also enables the development of AR applications in Unity that are easily portable to both platforms.

Although visual tracking now has the ability to recognize and track a lot of things, it mostly relies on other techniques such as GPS and accelerometers. For example, for a computer to detect and recognize a car it is very hard. The surface of most cars is both shiny and smooth and most of the feature points come from reflections and thus are not relevant for pose estimation and even sometimes recognition **[11]**. The few stable features that one can hope to recognize, such as the windows corners or wheels, are extremely difficult to match due to reflection and transparent parts. While this example is a bit extreme, it shows the difficulties and challenges faced by computer vision with most objects that have irregular shape, such as food, flowers, and most objects of art.

A recent new approach for advances in visual tracking has been to study how the human brain recognizes objects, also called the Human Vision System (HVS), as it is possible for humans to recognize an infinite number of objects and persons in fractions of seconds. If the way of recognizing things by the human brain can be modeled, computer vision will be able to handle the challenges it is currently facing and keep going forward.

1.2.2. AR devices

The main devices for augmented reality are displays, input devices, tracking, and computers.

1.2.2.1. Displays

There are three major types of displays used in Augmented Reality: head mounted displays (HMD), handheld displays and spatial displays.

HMD is a display device worn on the head or as part of a helmet and that places both images of the real and virtual environment over the user's view of the world (Fig 1.3). HMD can either be video-see-through or optical see-through and can have a monocular or binocular display optic. Video-see-through systems are more demanding than optical-see-through systems as they require the user to wear two cameras on his head and require the processing of both cameras to provide both the "real part" of the augmented scene and the virtual objects with unmatched resolution, while the optical-see-through employs a half-silver mirror technology to allow views of physical world to pass through the lens and graphically overlay information to be reflected in the user's eyes. The scene as well as the real world is perceived more naturally than at the resolution of the display. On the other hand, in Video-see-through systems, augmented view is already composed by the computer and allows much more control over the result. Thus, control over the timing of the real scene can be achieved by synchronizing the virtual image with the scene before displaying it while in an optical-see-through application, the view of the real world cannot be delayed, so the time lag introduced in the system by the graphics and image processing is perceived by the user. This results in image that may not appear "attached" with

the real objects they are supposed to correspond to, they appear to be unstable, jittering, or swimming around.

Figure 1.3 : Device HMD Microsoft Hololens on augmented reality [12].

Handheld displays employ small computing devices with a display that the user can hold in their hands (Fig 1.4). They use video-see-through techniques to overlay graphics onto the real environment and employ sensors, such as digital compasses and GPS units for their six degree of freedom tracking sensors, fiducial marker systems, such as ARToolKit, ARCore, Vuforia and/or computer vision methods, such as SLAM. There are currently three distinct classes of commercially available handheld displays that are being used for augmented reality system: smart-phones, PDAs and Tablet PCs [13]. Smart-phones are extremely portable and widespread, and with the recent advances present a combination of powerful CPU, camera, accelerometer, GPS, and solid state compass, making them a very promising platform for AR. However, their small display size is less than ideal for 3D user interfaces. PDAs present much of the same advantages and disadvantages of the smart-phones, but they are becoming a lot less widespread than smart-phones since the most recent advances, with Android-based phones and iPhones. Tablet PCs are a lot more powerful than smart-phones, but they are considerably more expensive and too heavy for single handed, and even prolonged two-handed, use. However, with the recent release of iPad, we believe that Tablet PCs could become a promising platform for handheld AR displays.

Figure 1.4: Handheld displays from mobile of an application Pokémon GO [62].

Spatial Augmented Reality (SAR) make use of video-projectors, optical elements, holograms, radio frequency tags, and other tracking technologies to display graphical information directly

onto physical objects without requiring the user to wear or carry the display (Fig 1.5) **[14]**. Spatial displays separate most of the technology from the user and integrate it into the environment. This permits SAR to naturally scale up to groups of users, thus allowing collaboration between users, increasing the interest for such augmented reality systems in universities, labs, museums, and in the art community. There exist three different approaches to SAR which mainly differ in the way they augment the environment: video-see-through, optical-see-through and direct augmentation. In SAR, video-see-through displays are screen based; they are a common technique used if the system does not have to be mobile as they are cost efficient since only off-the-shelf hardware components and standard PC equipment is required. Spatial optical-see-through displays generate images that are aligned within the physical environment. Spatial optical combiners, such as planar or curved mirror beam splitters, transparent screens, or optical holograms are essential components of such displays **[14]**. However, much like screen-based video see-through, spatial optical-see-through does not support mobile applications due to spatially aligned optics and display technology. Finally, projector-based spatial displays apply front-projection to seamlessly project images directly onto physical objects' surfaces, such as in **[15]**. More details about SAR can be found in **[14]**. Table 1.1 shows a comparison of different types of displays' techniques for Augmented Reality.

Figure 1.5 :SAR from Hands-On With the Iron Man Augmented Reality [16].

1.2.2.2. Input devices

There are many types of input devices for AR systems. Some systems, such as Reitmayr et. al.'s mobile augmented system [17]utilizes gloves. Others, such as ReachMedia [18] use a wireless wristband. In the case of smart-phones, the phone itself can be used as a pointing device; for example, Google Sky Map on Android phone requires the user to point his/her phone in the direction of the stars or planets s/he wishes to know the name of. The input devices chosen depend greatly upon the type of application the system is being developed for and/or the display chosen. For instance, if an application requires the user to be hands free, the input device chosen will be one that enables the user to use his/her hands for the application without requiring extra unnatural gestures or to be held by the user, examples of such input devices include gaze interaction in [19] or the wireless wristband used in [18]. Similarly, if a system makes use of a handheld display, the developers can utilize a touch screen input device.

1.2.2.3. Tracking

Tracking devices consists of digital cameras and/or other optical sensors, GPS, accelerometers, solid state compasses, wireless sensors, etc. Each of these technologies has different level of accuracy and depends greatly on the type of system being developed.

Table 1.1: Comparison of different techniques for different types of display.

Types of Displays Techniques	HMD		Handheld			Spatial		
	Video-see-through	Optical-see-through	Video-see-through			Video-see-throug	Optical-see-through	Direct Augmentation
			Types of Displays	HMD	Handheld			
Advantages	complete visualization control, possible synchronization of the virtual and real environment	employs a half-silver mirror technology, more natural perception of the real environment	portable, widespread, powerful CPU, camera, accelerometer, GPS, and solid state compass	portable, powerful CPU, camera, accelerometer, GPS, and solid state compass	More powerful	cost efficient, can be adopted using off-the-shelf hardware components and standard PC equipment	more natural perception of the real environment	displays directly onto physical objects' surfaces
Disadvantage	requires user to wear cameras on his/ her head, requires processing of cameras video stream, unnatural perception of the real environment	time lag, jittering of the virtual image	Small display	becoming less widespread, small display	More expensive and heavy	do not support mobile system	do not support mobile system	not user dependent: everybody sees the same thing (in some cases this disadvantage can also be considered to be an advantage)

In [20], the authors identified the general tracking technology for augmented reality to be: mechanical, magnetic sensing, GPS, ultrasonic, inertia, and optics. In [21].

1.2.2.4. Computers

The computer analyzes the sensed visual and other data to synthesize and position augmentations. Computers are responsible for the graphics that go with augmented reality. Augmented reality uses a computer-generated image which has a striking effect on the way the real world is shown. With the improvement of technology and computers, augmented reality is going to lead to a drastic change on one's perspective of the real world. [22], in about 15–20 years it is predicted that augmented reality and virtual reality are going to become the primary use for computer interactions. [23] Computers are improving at a very fast rate, leading to new ways to improve other technology.

1.2.3. AR interfaces

One of the most important aspects of augmented reality is to create appropriate techniques for intuitive interaction between the user and the virtual content of AR applications. There are two main ways of interaction in AR applications: tangible AR interfaces, collaborative AR interfaces.

1.2.3.1. Tangible AR interfaces

Tangible interfaces support direct interaction with the real world by exploiting the use of real, physical objects and tools. A classical example of the power of tangible user interfaces is the AR furniture application [24], which enables a person to select and rearrange the furniture in an AR living room design application (Fig 1.6).

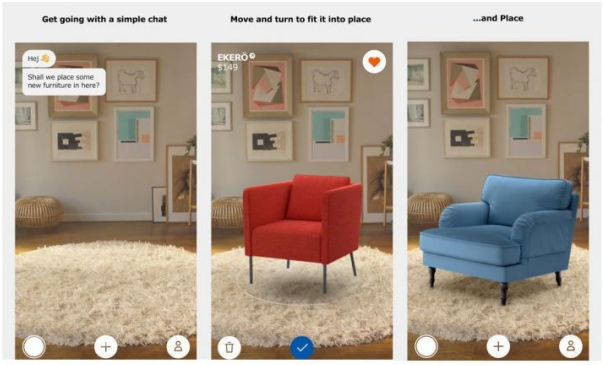


Figure 1.6: screenshot of AR furniture application [24].

1.2.3.2. Collaborative AR interfaces

Collaborative AR interfaces include the use of multiple displays to support remote and co-located activities. Co-located sharing uses 3D interfaces to improve physical collaborative workspace. In remote sharing, AR is able to effortlessly integrate multiple devices with multiple locations to enhance teleconferences. Remote sharing can be used for enhancing teleconferences such as in [25]. Such interfaces can be integrated with medical applications for performing diagnostics, surgery (Fig 1.7), or even maintenance routine.

Figure 1.7: Microsoft HoloLens used in Healthcare [26].

1.3. Applications

While there are many possibilities for using augmented reality in an innovative way, we have cornered four types of applications that are most often being used for AR research: advertising and commercial, entertainment and education, medical, and mobile application for iPhones. Below, we study why AR could bring a better solution to some areas, a cheaper solution to others, or simply create a new service. We also discuss the challenges augmented reality is facing to go from the laboratories to the industry.

Note that it was decided here to replace the navigational and informational domain application that was encountered in the AR systems section by a study of the augmented reality mobile applications as these applications most often have navigational and informational use Fig 1.3,1.4 and 1.5.

1.3.1. Advertising and commercial

Augmented reality is mostly used by marketers to promote new products online. Most techniques use markers that the users present in front of their webcam either on special software or simply on the advertising company's website. For example, in December 2008, MINI [27], the famous car company, ran an augmented reality advertisement in several German automotive magazines [28]. The reader simply had to go to the MINI's website [27], show the ad in front of their webcam, and a 3-D MINI appeared on their screen, as seen in Fig 1.8. Beyond Reality [29] released a marker less advertisement magazine of 12 pages that could be recognized and animated by a software the user could download on the publisher's website as a starting point to their Augmented Reality Games. They see that with such a system, they could add a "paid" option on the software that would allow the user to access additional content, such as seeing a trailer and then being able to click on a link to view the full movie, turning the magazine into a movie ticket [29].

AR also offers a solution to the expensive problem of building prototypes. Indeed, industrial companies are faced with the costly need to manufacture a product before commercialization to figure out if any changes should be made and see if the product meets the expectations. If it is decided that changes should be made, and this is more often than not the case, a new prototype has to be manufactured and additional time and money are wasted.

Figure 1.8: MINI advertisement [28].

A group of the Institute of Industrial Technologies and Automation (ITIA) of the National Council of Research (CNR) of Italy [30] in Milan works on AR and VR systems as a tool for supporting virtual prototyping. The ITIA-CNR is involved in the research for industrial contexts and application using VR, AR, Realtime 3D, etc. as a support for product testing, development and evaluation. Some examples of applied research projects where the above technologies have been applied include motorcycle prototyping (Fig 1.9), virtual layout of a factory and an office (Figs 1.10 and 1.11), virtual light simulation (Fig 1.10), and virtual trial of shoes with the Magic Mirror interface, which will be discussed next (Fig 1.12).

Figure 1.9: right picture: Picture of a virtual motorcycle prototype (on the right) next to a physical motorcycle prototype (on the left) in the real environment; left picture: virtual prototype in a virtual environment [30].

Figure 1.10: Virtual factory prototype [30].

Figure 1.11: Virtual office prototype [30].

Figure 1.12: Virtual testing of alighting system [30].

Similar examples to Magic Mirror use of AR for advertising and commercial applications lies in fully replacing the need to try on anything in stores, thus saving considerable amount of time for clients, which would most likely be used for trying on more clothing (shirts, dresses, watches, pants, etc.) and thus increasing the stores chances for selling.

Augmented reality has not fully reached the industrial market in advertisement application mostly because a few improvements need to be made to systems similar to the Magic Mirror (Fig 1.13) or Cisco's retail fitting room (Fig 1.14).

Figure 1.13: User trying on virtual shoes in front of the Magic Mirror [30].

Figure 1.14: Cisco's AR commercial where a client is trying on clothes in front of a "magic" screen.

Indeed, for the product to be viable in the market it needs to provide the user with a flawless representation of the prototype; the user should have the impression that they are looking at a physical prototype. In the case of the Magic Mirror system, this would mean flawless tracking so that when the user looks at the magic mirror s/he feels like s/he is actually wearing the shoes and can really see what the shoes would look like.

1.3.2. Entertainment and education

Entertainment and education applications include cultural apps with sightseeing and museum guidance, gaming apps with traditional games using AR interfaces, and some smart-phone apps that make use of AR for an entertainment and/or educational purpose.

In cultural application, there exists a few systems that uses AR for virtually reconstructing ancient ruins (Fig 1.15), such as in [6], or for virtually instructing user about site's history such as in [31].

Figure 1.16: Augmented view of Dashuifa from [6].

Figure 1.15: Mobile phone-enabled Mobile phone-enabled guidance in a museum from [32].

There are also a few systems that exploit AR for museum guidance such as [32] and [33]. In [32] and [33], both systems are mobile, but [32] also uses a mobile phone as an interface (Fig 1.16) while [33] simply uses a magic lens configuration (Fig 1.17). In [32], the authors identified the benefits of using augmented reality as an interface for their cultural applications as: efficient communication with the user through multimedia presentations, natural and intuitive technique and low maintenance and acquisition costs for the museum operators' presentation technology in the case of smart-phone being used as an interface. And indeed, using a smart-phone or even another hand-held display is a more intuitive and natural technique than looking up a number randomly assigned to the object in a small written guide, especially when the user can simply make use of his/her own phone in a world where everybody already possesses one. Similarly, users can relate easier to multimedia presentations brought to them and will more willingly listen, watch and/or read about information that they can acquire by simply pointing at an object using their phone rather than have to look it up in a guide.

Figure 1.17: Visitor with guidance system from [33].

1.3.3. Medical applications

Most of the medical applications deal with image guided and robot-assisted surgery. As a result, significant research has been made to incorporate AR with medical imaging and instruments incorporating the physician's intuitive abilities. Significant breakthrough has been provided by the use of diverse types of medical imaging and instruments, such as video images recorded by an endoscopic camera device presented on a monitor viewing the operating site inside the patient. However, these breakthroughs also limit the surgeon's natural, intuitive and direct 3D view of the human body as the surgeons now have to deal with visual cues from an additional environment provided on the monitor [34]. AR can be applied so that the surgical team can see the imaging data in real time while the procedure is progressing. [34] introduced an AR system for viewing through the "real" skin onto virtual anatomy using polygonal surface models to allow for real time visualization (Fig 1.18). The authors also integrated the use of navigated surgical tools to augment the physician's view inside the human body during surgery.

Figure 1.18: Bichlmeier et. al. system for viewing through the skin [34].

1.3.4. Mobile applications

This is the legendary project by Niantic which brought Augmented Reality games into the crowds. The nostalgia of players enabled Pokémon GO to hold several Guinness records and gain incredible earnings. The role of this AR game in terms of making Augmented Reality technology known to millions of people is hard to overstate [35].

Pokémon GO is an adventure type of mobile game (Fig 1.18), which places the battlefield on the real surroundings and superimposes virtual animate objects into user's environment. Millions or maybe even billions of Pokémons have been caught in years since 2016. And the fun still continues [35].

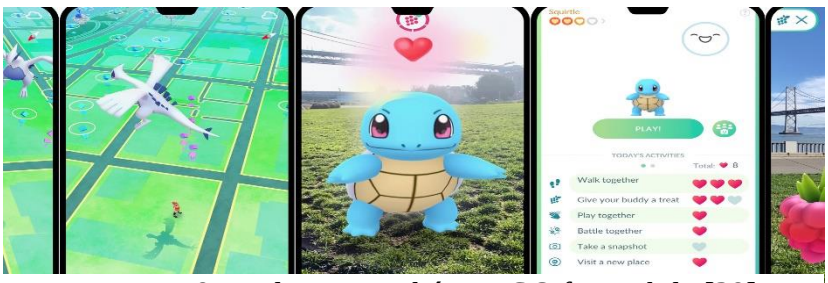


Figure 1.19: application Pokémon GO for mobile [39].

There are other examples of such applications include WikitudeDrive (Fig 1.20) [36], which is a GPS-like application that allows the user to keep his/her eyes on the road while glancing at the GPS, and Restaurant Guide (Fig 1.21) that has a navigational function to guide the user to the nearest restaurant that serves food.

Figure 1.21: WikitudeDrive [36].

Figure 1.20: Restaurant Guide [37].

1.4. Conclusion

Augmented reality (AR) systems aim at adding virtual objects to real scene to make the virtual objects merge with the existing world seamlessly in such a manner as to appear part of the viewed 3D scene. In contrast with virtual reality, users can see virtual objects and the real world simultaneously in the augmented reality system.

To enhance the illusion that the virtual objects are actually present in the real scene, researchers have paid more and more attention to the occlusion problem.

Through the next chapter, we will present Methods handling occlusions.

Chapter2

2. handling occlusions : theoretical part

2.1. Introduction

In the augmented reality systems, virtual objects are usually rendered on the video image without using depth information from the real scene, so real objects are always occluded by virtual objects. We call this the "occlusion problem", as shown in Figure 2.1. This problem results in poor understanding of the geometrical relationship between real and virtual objects.

2.2.Occlusions in augmented reality

The problem of occlusion occurs when the real objects are in front of the virtual objects in the scene. Without occlusion handling, users will have the misconception that the real object is further from the viewpoint than the virtual objects when the virtual objects are occluded by the real objects in the scene [38]. An example of an occlusion problem is shown in Fig 2.1. In Fig 2.1, the virtual object is a target. Fig 2.1. (a) is the composed image looking down from the top of the scene. We can find that the target is behind the red caddy if we see from the viewpoint as shown in Fig 2.1 (c). This means that part of the target should be occluded by the red caddy. Fig 2.1 (b) is the composed image overlaid with virtual target without occlusion handling. The image gives people the impression that the target is in front of the caddy, which is not the case in fact.

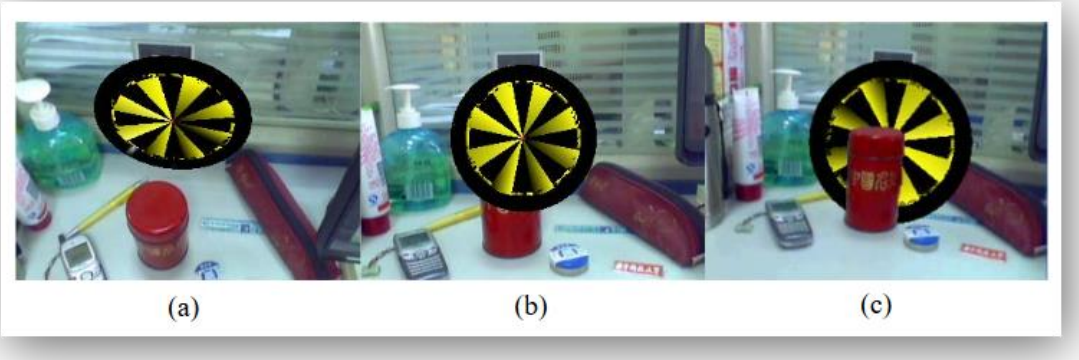


Figure 2.1: Occlusion problem in augmented reality.

Various approaches have been suggested for handling occlusion problems in augmented reality. They are mainly classified into two types: model-based and depth-based approaches. We can see the classification of these methods in the table 2.1.

Table 2.1: classification of methods handling occlusions in augmented reality.

Classification	Method	Camera type	3D modeling of the RE
model-based	Photogrammetry for 3D reconstruction	Digital camera	yes
depth-based approaches	Real-time 3D reconstruction	Depth camera	no

2.3.Photogrammetry for 3D reconstruction method

2.3.1. Definition

Photogrammetry is the science of making measurements from photographs. It infers the geometry of a scene from a set of unordered photographies or videos. Photography is the projection of a 3D scene onto a 2D plane, losing depth information. The goal of photogrammetry is to reverse this process. 3D reconstruction from multiple images is the creation of three-dimensional models from a set of images Using a technique Photogrammetry.

2.3.2. System overview

The Science of calculating the geometrical data in terms of size, shape and position of objects by reviewing images/photographs captured using electronic medias like digital cameras or mobile phones with image capturing capabilities. The fundamental objective of photogrammetry methodology is to connect geometrical data of image plane and 3D object in space while imaging. After correctly establishing this relationship, the corresponding geometrical data and its physical quantities of that object could be gathered from those images. Photogrammetric method could be a game-changer in particular cases where objects those need to be measured are complex or hard to access, moving objects or deformations and their surface and/or contour data is required. According to Slater, [39] there are limitations to the human eye. In terms of logistics, out of current computer vision/visualization devices, photogrammetry seems to be best promising method. Photogrammetric tools extract accurate geometric properties and configurations of objects from overlapping pictures and automates to generate 3D content by triangulation method of common visible points from different photographs. Triangulation is a mathematical method for calculating intersections and lines in order to extract 3D coordinates. After complete calculations, one is left with close digital model obtained from the photographs [39]. To obtain necessary data about 3 space coordinate system at least 2 projections are necessary which can be derived from photographs. Rays (or lines of sight) are drawn from the position of the camera to point of interest (POI) and the mathematical equations are used for production [40]. From 2 photographs of the same objects, its original true dimension could be calculated to build 3D model [41]. Process involving multiple photographs requires high-end CPU and GPU with effective reconstruction software. If positions and directional locations of camera are known, then rays can be mathematically met to get xyz coordinates of the POI (see Figure 2.2) [40].

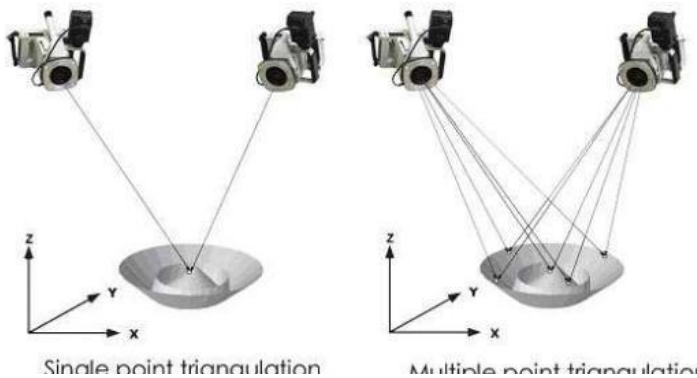


Figure 2.2: Single point and Multipoint Triangulation [41].

A bundle triangulation happens when numerical convergences are simultaneously calculated for all spread out photographs. Multiple images generate many rays or lines of sight. The estimated xyz coordinates are located in local coordinate system. As the principal use of post-processing software tools is to create 3D model. But, these softwares are not able to handle scale factors and transformations (see Figure 2.3) [40].

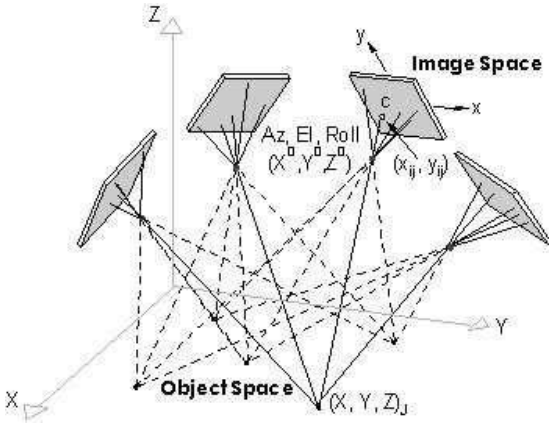


Figure 2.3: Multiple POI's from multiple camera images [41].

Photogrammetry is a part of image-based modelling category, a group which con-sists exotic technologies like shape from shading, shape from silhouette and shape from texture [42].

Previously custom special instruments for photogrammetric measurements were needed for those whoever planning a 3D reconstruction project. Due to more significant degree of automation capability, standard computing equipments are used. Moreover, expert level skills are not necessary in order to carry out project work and processing. Look at flowchar suggesting reduction in total project time after evolution of digital cameras [40].

2.3.2.1. Classification of photogrammetry

Photogrammetric projects are generally conducted using camera devices like mobile phones, Digital Single Lens Reflex (DSLR) cameras or drones/unmanned aerial vehicles (UAV).

Quality of images varies as resolution of cameras are different. Thus, these devices may require to record more photographs [43]. Photogrammetry techniques can be categorized as shown in the Fig. 2.4 [44].

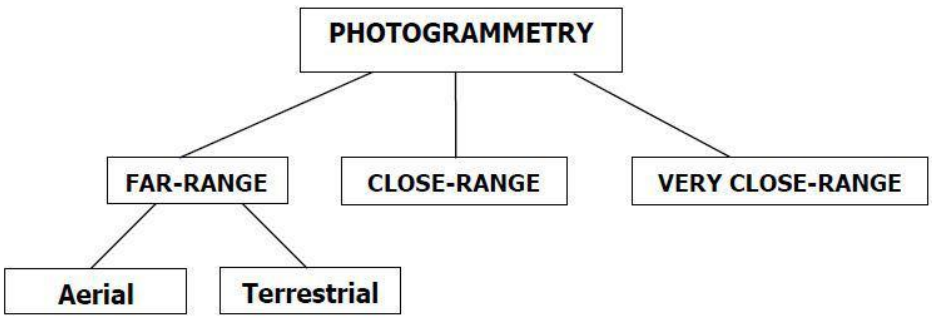


Figure 2.4: Classification of Photogrammetry methods [45].

Far-range Photogrammetry: is applied to reconstruct the shape of building, to create topographical maps etc. [44] Range: (areas of m2). Aerial photogrammetry is conducted using UAVs to which a camera is attached for capturing images while it's in air [43].

Close-range and very close-range Photogrammetry: generally used for industrial applications where non-contact measurements has to be done. Range: could vary between (1 cm 2to 1 mm 2) [44]. For Close-range photogrammetry, photographs are captured within 300 m of test object using DSLR cameras/ cellphone [43].

2.3.2.2. Characteristics of Close-range Photogrammetry

According to Jing [45], following are the characteristic of close-range photogrammetry.

- High Measurement Accuracy
  - — Using Single camera measuring system, relative accuracy up to 5 m can be achieved. For high configuration, it could be 5 m / m [45].
- High Efficiency
  - — Within concise period, feature information is extracted from tens of thou-sands of data points [45].
- Stable Performance
  - — Test objects have unaffected high precision measurements quality when used in adverse environment like high temperature, hazardous environment, high pressure and electro-magnetic environment [45].
- Non-contact Measurement
  - — So without touching or doing any damage to object high accuracy measurements can be done [45].

2.3.3. Photogrammetry used for Augmented Reality

The current hot topic around the world is AR. This topic has inspired many researchers to create algorithms for effective object reconstruction. For facility-management application in the construction industry, these AR model can play an important role. Also, for teleoperating which means interaction with machinery, autonomous vehicles and robots virtual models are needed in mining industry [46]. However, the gaming industry is booming at a much higher rate than ever before, have shown significant interest in developing advanced reality games. Even entertainment industry is not lagging behind using virtual models created by object reconstruction. As real-life models are getting more demand than artificially produced content, creating AR objects does not inherently need any measurements to be made. They can be modeled from right from scratch. Just because of the evolution of higher computing and graphics processing rates, real-world data can potentially be generated as realistic looking models in a more automated and quicker way [47]. In many cases, original dimensions with real-life objects texture are often combined with artificial texture by partial graphical manipulation to get an appropriate virtual model. In cases where only remains of the monuments are present but entire structure of that archaeological site has to be recreated in augmented reality.

2.3.4. Photogrammetry Test Setup

As mentioned earlier, Photogrammetry uses many images to crack the collinearity equations of collected data. In close-range photogrammetry, image matching is processed based on stereo-pairs. Photogrammetry software needs parallel images with at least 60% overlap. Also, photogrammetry involves control points in order to scale and recognize features. The position of the points must be identified for accurate data investigation to generate the 3D model [43]. Thus, capturing photographs of the object is the crucial stage of entire process.

2.3.4.1. Acquisition of Images

During image acquisition process, it is important to know whats need to be measured and reconstructed using techniques like photogrammetry and dense matching. Along with that, an appropriate contrast and triangulation of surface points for photogrammetry should be considered. Thus, it is recommended to adjust within a correct angle of triangulation and an extreme variation of perspective views between successive photographs [48]. Author suggests to avoid high angular views and recommends angular views should range from 60 to 110 degrees (i.e. consecutive photos should have 20 to 30 degrees of difference). See figure 2.5. The accuracy of the photogrammetric depends on the basis of relative angle between pairs of images [48]. For photogrammetry from small to large scale objects image acquisition method changes. Figure 2.5 shows how to capture small scale object and Figure 2.6 shows how to capture large-scale object for reconstruction [43]. Based on the scale of an object to be modeled, correct method of photo capturing changes [43].Generally images are captured around the object in clock-wise direction and combining dissimilar heights and view angles for each image acquisition position [48].

While acquiring small objects, depending on the shape and geometry of the object, a polar coordinate method for location of camera can be used. In this configuration, minimum two process parameters need to choose. One could be tilt angle and another could be step angle. As shown in the figure 2.7, Tilt angle (Ψ) denotes the tilt of the camera with respect to the xy plane of the object. And the step angle (Θ) is the rotating phase of the turning table. This generally controls the number of images and the overlying degree between two successive images. These are the two important factors which decides image capturing approach [49].

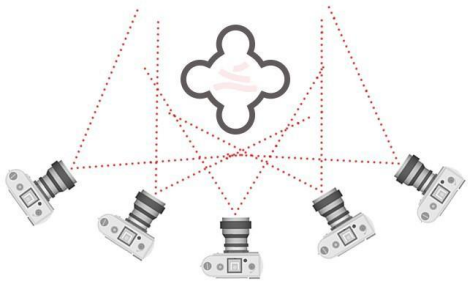


Figure 2.5: Image acquisition method for small objects' reconstruction [43].

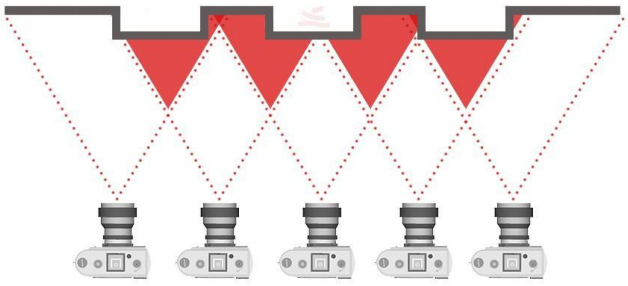
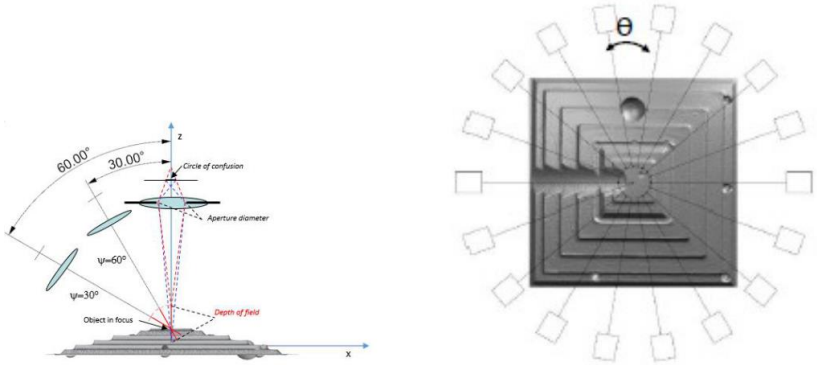


Figure 2.6: Image acquisition method for large objects' reconstruction [43].



(a) Tilt angle (Ψ) (b) step angle (Θ)

Figure 2.7 Camera Positions Strategy [25].

2.3.4.2. Setup Instructions

White cloth can be placed around the object scene to avoid detecting arbitrary points that could be processed as targets. This helps in the generation of the 3D point cloud data by avoiding unwanted data from the backgrounds of the object. This will reduce undesirable noise as well as the total time to perform the calculations of the point cloud and final mesh, as there are less points to consider in triangulation [39]. Use of tripod as stabilizer, is recommended in order to get desired sharpness of images. The scene should be properly illuminated with di used light so as to avoid light reflections from object and dark shadows. If object surface is shiny, special removable matting powder could be sprinkled over objects body before taking images [50].

2.3.4.3. Factors Affecting Close-range Photogrammetry

The accuracy of obtained reconstructed model from a photogrammetry measurement significantly relies on the many associative parameters that are tangled in the photogrammetric process. According to braybon [40], The most dominant factors contain following:

- Camera and Lens Quality: the resolution of the cameras has a substantial role in precisely locate the position of a targets and markers [40]. The accuracy is usually improved with smaller object pixel size. This Pixel means the smallest box size which stores image data of the captured object using corresponding camera. The object pixel size is affected by the megapixel of the device [51].
- Sizes of Objects: Smaller object sizes generally result into higher accuracy of reconstructed model.
- Number of Images: This is probably the most dominant factor defining the accuracy of models. Increasing the number of overlapping images enhances the accuracy of the output 3D model. On the other hand, capturing the coded targets in more than four images does not improve results significantly [51].
- rea of focus of Images: Consecutive photographs taken with broader angles, result in the higher the accuracy of the recognition. As discussed earlier, angle of intersection should not be less than 60°. The ideal angle of intersection would be 90° [40]. See figure 2.8 illustrating accuracy pyramid and factors' influence on reconstructed model accuracy.

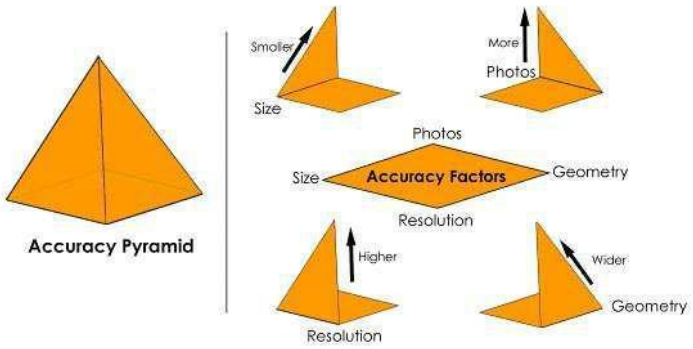


Figure 2.8: Accuracy Pyramid for Photogrammetry [40].

2.3.4.4. Aspects for Successful Photogrammetry

According to Luhmann [52], to implement photogrammetry successfully in the industry, lot of technical tools and components are desired to provide affordable and efficient system. Here is the list which summaries required tools and corresponding technical troubles:

- Cameras for imaging resolution (number of pixels), available lenses, acquisition and data transfer speed, camera stability, synchronization, data compression, etc.
- Target and Illumination representation of object features, target shape and dimensions, the wavelength of light devices, permissions to touch object, illumination power and measurement volume.
- Imaging Configuration number of camera stations, desired measurement ac-curacy, network design, redundancy, robustness, self-calibration ability, datum definition and object control, self-control of orientation and calibration.
- Image Processing automation of target recognition and identification, sub-pixel measurement of target center, multi-image matching approaches, feature tracking, and handling of outlines and scene artifacts.
- 3D model Reconstruction methods for determination of 3D coordinates (e.g., spatial intersection, bundle adjustment) and error statistics.
- Data interfaces integration into CAD/CAM environments, machine and data interfaces, user interaction and displays, etc.
- Verification for Accuracy reference bodies, reference data, standards and guidelines, and acceptance tests [52].

With above mentioned aspects, author says close-range photogrammetry system is a series of complex procedure which include proper design, setup and operation. The applicability of a method depends on technical issues as well as a function of required cost-performance ratio, system assistance, documentation, quality assurance and interdisciplinary skills. Due to this, there are less than 10 professional companies around the globe those provide a system in photogrammetry field probably. Though, the market for optical 3D measurements is significantly increasing and bids promising scenarios for the future [52] . Which Camera to use for Photogrammetry. In [43], author summarizes comparison between professional digital cameras and cellphone cameras. For Close-range photogrammetry method, cellphone camera or digital cameras are used. Cellphones are moderately cheap and need fewer know how to control. Cellphones have built-in sensors which can collect 3D locations and high-resolution photographs. Cell phones

are global and easily accessible to the research public. Nowadays cellphones are as accurate as metric cameras which not only save time but money also. When calibrated, mapping with cellphone devices enhances accuracy [43]. In [53], author has inferred that cellphones used in close-range photogrammetry are: fast and useful, efficient, higher accuracy compared with high-resolution Digital cameras. Other researches have concluded that cellphones are adequately accurate for small-scale projects that don't need high accuracy. Today's camera cellphones are proficient enough for creating high-resolution models as there is a lot of technological advancement in this area [43] .Studies predict that cellphones and digital cameras (DSLR) provide almost similar results for 3D construction. However, digital cameras are cost-effective and time -saving which can provide high-quality 3D CAD model. The primary difference be-tween DSLR cameras and cellphones is accessibility and image quality. Cell phone acquires a compressed image (JPEG), while the DSLR captures a RAW uncompressed image. Compression of a photo reduces quality [43].

2.3.5. Photogrammetry methodology and tools evaluation

If you own the most important tool which is a camera (could be DSLR or cellphone), then it is incredibly affordable to use close-range photogrammetry. Along with this, you need a photogrammetry toll to reconstruct a 3D object from the photographs you have captured [54]. Similar to other things, photogrammetry software is available with many extra features and functionality. Developers from industry have published commercial solutions for photogrammetric methodologies which could be useful in industrial and manufacturing applications. Though, there are some programs offered for free down-load [53]. In this part, based on website and blogs, the following section is discussing available commercial, professional packages and open source codes used to obtain 3D CAD models from images. Moreover, we discuss which free photogrammetry software tools. See table 2.1 for Photogrammetry tools available in the market [53], [55].

2.3.5.1. Software Evaluation

In this previous section, introduction about free open source codes for photogrammetry has been made. Now, it's time to evaluate these softwares based on common example and factors associated with. As shown in the table 2.2, all of these photogrammetry software packages produce a 3D data set of points, which is referred as 3D point cloud. From this list of packages, most of the programs generate triangulated surfaces using previously created point cloud. Generally, in order to get maximum bene t out of these open source packages in any research applications, the software program should be associated with preview window. This can show the photographic images with the created surfaces. According to [54], not all of these open source codes generate point cloud data. Packages like COLMAP, VisualSFM and OpenMVG are specifically programmed to return output in 3D point cloud with discrete points. Other tools like Meshroom, MicMac and Regard3D generate optimized 3D surface meshes from the point cloud data. Another method used for photogrammetry projects to generate the 3D surface is by exporting 3D point cloud data into separate 3D CAD system (3rd party application like - MeshLab) to generate surface mesh as per users' requirement to corresponding projects. The 3rd-party CAD system could also be used not only for manipulation, measurement the surface but also for determining the orientation of planar features [56]. According to shaffner [56], author has discussed difficulties faced by researchers and complexities associated with exporting data sets, surfaces to another CAD plat-form for analysis. Experienced CAD users

around the world using high-end computer systems, this method is generally preferred or acceptable. Thus, here comes the evaluation criteria for mentioned software packages are included in the following considerations:

- Efficient image processing procedure to generate 3D object reconstruction data.
- Minimal operation time and cost
- Data export capabilities to other CAD systems
- Built-in data post-processing tools (or is additional software needed)

During this photogrammetry research, referring table 2.2, (Regard3D, Meshroom and MicMac) three open source codes have been selected for evaluation.

Table 2.2:Photogrammetry solutions available in the market (year 2019) [55], [56].

Name	OS	Price	Interface	Vendor	Surface
Bentley ContextCapture	Windows	Paid	GUI	Bentley	YES
iWitnessPro	Windows	Paid	GUI	Photometrix	YES
Photomodeler	Windows	Paid	GUI	Photomodeler Technologies	YES
Autodesk ReCap	Windows	Paid	GUI	Autodesk	YES
RealityCapture	Windows	Paid	GUI	CapturingReality	YES
Metashape	Windows, macOS, LINUX	Paid	GUI	Agisoft	YES
3DF Zephyr	Windows	Paid	GUI	3DFLOW	YES
COLMAP	Windows, macOS, LINUX	Free	GUI & CLI	COLMAP	NO
Meshroom	Windows	Free	GUI	Alicevision	YES
MicMac	Windows, macOS, LINUX	Free	CLI	MicMac	YES
Regard3D	Windows, macOS, LINUX	Free	GUI	Regard3D	YES
VisualSFM	Windows, macOS, LINUX	Free	GUI	VisualSFM	NO
OpenMVG	Windows, macOS, LINUX	Free	CLI	QGIS	NO

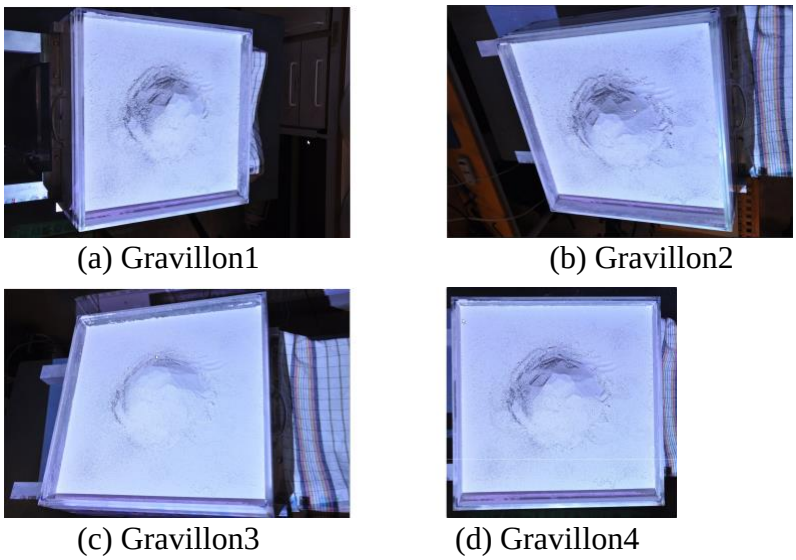
2.3.5.2. Selection of Tool for Integration

Selection criteria for photogrammetry tools were based on the cost of tool and complete work-ow until surface reconstruction. These categorised tools are known as Regard3D, Meshroom and MicMac. For evaluation, a common photoset "Gravillon" is used and then using tutorial documentation results were obtained using each photogrammetry tool. This "Gravillon" dataset

is light by design which has 4 images. This data-set is explicitly designed for beginners with light photogrammetry background by MicMac engineers. L.Girod developed Gravillon dataset who is from the University of Oslo, Norway. This dataset was captured to model a volcano created by O.Galland. See figure 2.9 where this "Gravillons" dataset has been displayed [57].

Regard3D, Meshroom and MicMac have their own advantages and disadvantages. All of these open source codes provide information about all steps of photogrammetry calculation and supply statistical testing of the derived parameters.

On the point note, MicMac does not create intermediate les of different procedure which helps saving storage space in the user's system. During cloud compare analysis, it's been discovered that MicMac has generated lesser number of points in the cloud than Meshroom and Regard3D (Refer Fig 2.10). Also, when scales are converted in 0 to 1, the area under curve for MicMac in graph is standing last. Thus, the model reconstructed is inferior in quality. It took more time than other tools, as MicMac is based more on CPU than GPU. There is no doubt about changing parameters in the tool's setting in MicMac, but this program is hard-coded which makes it di cult to manipulate. In [55], the author has rated various photogrammetry software on qualitative and user compatibility basis and MicMac scored 2 out of 10 which makes it last in standings. Henceforth, eliminating MicMac from integration consideration would be a logical decision.



(a) Gravillon1 (b) Gravillon2

(c) Gravillon3 (d) Gravillon4

Figure 2.9: Gravillon Dataset by MicMac [58].

Meshroom needs less post-processing operation than Regard3D to achieve approximately correct reconstructed object from the same set of images. Another deciding factor for integration is the time of operation for the photogrammetric method. Meshroom has produced a moderately better quality of object within less time of computational time of 1 min 46 sec to Regard3D's 4 min 53 sec. In [55], the author has summarized about troubles getting correct, appropriate results with Regard3D and rated Regard3D 5 out of 10. During this evaluation, Meshroom has come out as a simple but versatile tool which has options between automatic and advanced settings. Meshroom photogrammetry tool has several nodes in its pipeline. Each of those nodes could be altered as per requirement. Also considering, computational time for same simple dataset, ability to match scan data Meshroom is better in all 3 photogrammetric

tools discussed. Therefore, in this chapter, the methodology of Meshroom photogrammetry tool has been studied.

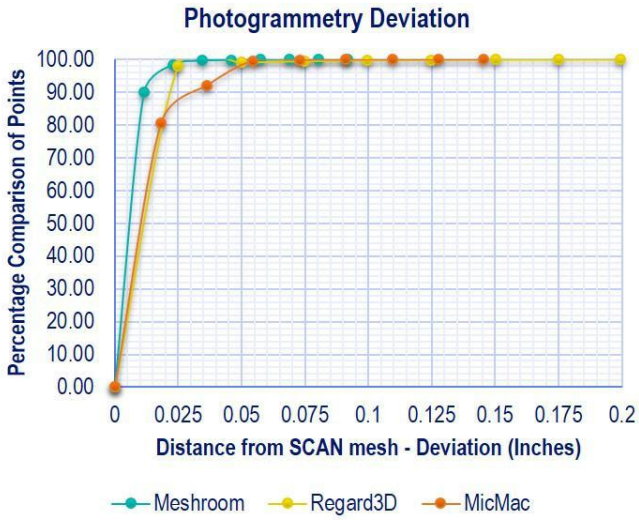


Figure 2.10: Comparison of c2c deviation of photogrammetry tools.

2.3.6. Photogrammetry algorithm of Meshroom

As discussed in the previous chapter, Meshroom is built on Alice vision framework with a simple node-based methodology which connects necessary steps to reconstruct images to the 3D object. The Default pipeline of Meshroom has a total of 10 nodes which results in a complete reconstructed 3D model from its images, shown in the figure 2.11. Let's walk through these default nodes of the photogrammetry one by one -As per Alicevision Meshroom' s documentation [59].

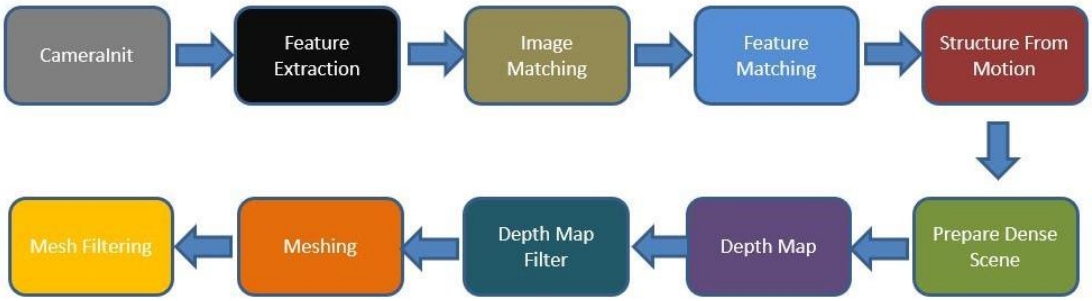


Figure 2.11: Meshroom Photogrammetry Pipeline.

2.3.6.1. CameraInIt

- The very first step generates a (.SFM) file. For photogrammetry, camera/sensor is essential equipment to record images. Every camera has parameters like camera/sensor type, focal length, size of images can be captured and other parameters.
- While using photogrammetry tool, it is recommended to store camera in- formation or data which can be used for custom camera calibration in later stages. Thus, in this CameraInit stage, a file that could store all necessary data about a variety of cameras need to be stored in an array of matrix format.

- In computing, JSON (JavaScript Object Notation) is an open standard file format that utilizes human-readable text to transmit data objects consisting of attribute-value pairs and array data types (or any other index value) [60].
- SFM files are JSON files that store Camera Size, Sensor information, distortion coefficient camera extrinsic matrices, bundle points.

2.3.6.2. Feature Extraction

- The main objective of this node in photogrammetry is to extract characteristic sets of pixels that are, to some level, unaffected to altering camera viewpoints throughout image capturing. Therefore, a feature in the image scene should have similar descriptions of feature in all images.
- The most familiar feature recognition method is the SIFT (Scale-invariant feature transform) system. Regardless of rotation, scale and translation, SIFTs preliminary motto is to identify discriminative pixel sets in the first photo which could be compared with discriminative pixel sets in the second photo.
- The extracted pixel sets are centered at stable points of interest as this is a relevant aspect that only occurs at a specific scale. Keeping this in mind, the gist of the algorithm is that to some extent one could be benefited from SIFTs invariance property to address image transformations which occur when viewpoints of the camera are changed during image capturing procedure.
- By computing a pyramid of scaled-down images, one image can be represented at various scales. Scale-space maxima of the Laplacian representation is calculated by the SIFT method. Laplacian representation is a precise image energy-based representation of the image. SIFT uses nominated Gaussian differences. Point of interest generally linked by these maxima. It then samples for each one of these maxima a square image patch whose origin is the maximum and x-direction is the dominant gradient at the origin. Detailed information is linked to each keypoint [59].
- The detailed description is characteristically stored in 128 bits. This de- scription contains statistics of gradients calculated in pixel patches about the keypoint. Region size is determined by the keypoint scale while the orientation is determined by the principal axis determines [59].

2.3.6.3. Image Matching

- After feature extraction by SIFT method, in this step, the objective is to match images that observing similar parts of the scene. In this, image pair is created by image recovery methods to find images that share nearly the same information without resolving every feature match in minutiae.
- The objective is to restructure the image in a dense image descriptor which calculates the distance between all dense images descriptors proficiently.
- The vocabulary tree approach is one of the most common technique to create this image descriptor. Once all extracted features descriptors inserted into this method, it categorizes their descriptors after comparison to the ones on every node of this tree [59].
- Every leaf on this tree is associated with one feature descriptor. The index of conforming leaf can store this feature descriptor in the tree. Then, the image descriptor is signified by this group of denoted leaves' indices [59].

- It is now conceivable to check if different images share similar information by comparing these image descriptors with others.
- The output feature extraction folder(s) and descriptors are inserted as input to Image matching node. During this process, a vocabulary tree file (.tree) provided in Meshroom's GitHub folder can be used for indexing purposes. When the user computes node by node results, weight file can be given a name; otherwise, the weights are computed on the database built with the provided set [59].
- To get an accurate 3D model from image reconstruction, a higher number of image capturing is advised. In this image matching step, a minimal number of images to use the vocabulary tree can be set between 0 to 500 images. Any number of images less than the set threshold parameter are computed for matching combinations.
- Along with this parameter, the number of descriptors user load per image can be limited and retrieval of the number of matches can be set.

2.3.6.4. Feature Matching

- The goal of this node is to match all features between appropriate and qualified image pairs.
- Firstly, this node accomplishes photometric matches between the set of descriptors using 2 input images. For every feature in image A, a list of contender features in image B is obtained. To remove bad matching candidates, Meshroom assumes that theres only one right match in the other image [59].
- Thus, for each feature descriptor on the 1st photo, this step looks for the two nearby descriptors and uses a relative threshold between them. This postulation eliminates features on repetitive assembly but has exhibited to be a robust standard [59].
- This node gives an output of a list of feature matching contenders validated by only a photometric criterion. Identifying two closest descriptors in a 2nd image for every feature is intense for computation with the brute force method; though many researchers have optimized this existing algorithm [59].
- This node in Meshroom then uses the features positions in the images to make geometric filtering by using Epipolar geometry in an outlier detection framework called RANSAC (RANdom SAmple Consensus) [59].
- Later, this node arbitrarily selects a minor set of feature correspondences and compute the fundamental (or essential) matrix; then it checks the number of features that authenticate this model and iterate through the RANSAC framework [59].

2.3.6.5. Structure from Motion

- The goal of this node is to comprehend the geometric relationship associated with every observation provided by the input images and conclude the rigid scene structure with 3D points, their position, their orientation and internal calibration of every camera.
- The Incremental pipeline is an upward reconstruction procedure. Firstly, this node calculates an initial two-view reconstruction and then it is iteratively stretched by introducing other views.
- Further, it combines every feature match between image pairs into a track. Each track is assumed to characterize a point in space, noticeable from numerous cameras. At this

node of the pipeline, input still comprises many outliers. During this fusion of matches, this node also removes dis- jointed tracks [59].

- Then, the incremental algorithm has to select the top matching initial pair of images. This selection is vital for the superiority of the final reconstruction of the object. It should indeed deliver robust matches and comprise reliable geometric data. Also, this image pair should capitalize on the number of matches and the repartition of the conforming features in every image. Nevertheless, at a similar time, the angle between the following camera positions should also be big enough to offer reliable geometric data.
- Then this step calculates the fundamental matrix between these two photos and assumes that the 1st one is the origin of the coordinate system. Now, Meshroom registers position and orientation of 2nd camera and tri- angulates conforming 2D features into 3D points [59].
- Further, Meshroom node selects every image that has adequate relations with the features that are already reconstructed in 3D space. Based on these 2D-3D relations, this SFM node completes the resectioning of every new camera. The resectioning is a Perspective-n-Point algorithm (PnP) in a RANSAC framework to identify the position and orientation of the cam- era device that validates most of the features relations. On each camera, a non-linear minimization is completed to refine the pose [59].
- From these new cameras positions, some tracks become observable by two or more resected camera devices and node triangulates these positions. Then, node launches a Bundle Adjustment to improve the whole thing: extrinsic and intrinsic parameters of all camera devices as well as the position of all 3D points. Meshroom SFM node filters Bundle Adjustment results by removing every observation that has large reprojection error or inadequate angle between viewpoints [59].
- After triangulation of new points, more candidates are available for the next best scene view selection. Bundle adjustment iterates this by adding camera poses and triangulates new 2D features into 3D points and eliminates 3D points which had become invalidated until node can’t find a new view [59].

2.3.6.6. Prepare Dense Scene

- “Prepare Dense Scene” node’s primary function is to undistort images. This step produces undistorted EXR photos which eliminate projection conversion steps and depth calculation back and forth from the distortion function.

Figure 2.12: Prepare Dense Scene [60].

2.3.6.7. Depth Map Estimation

- For every camera that has been determined by SFM, this node retrieves the depth value of each pixel of the image. To estimate depth value, numerous approaches like Block Matching, Semi-Global Matching (SGM) or ADCensus exist. This node focuses on the SGM method executed in AliceVision Meshroom pipeline [59].
- For every image, the node selects the N best/closest camera devices nearby. This node selects front-parallel planes based on the intersection of the optical axis with the pixels of the selected neighboring cameras. This node produces a volume W, H, Z with numerous depth contenders per pixel of the image. This node estimates the similarity for each one of the pixels. The similarity is calculated by the Zero Mean Normalized Cross-Correlation (ZNCC) of a minor area in the main photo reprojected into the other camera device [59].
- This node generates a volume of resemblances. For every neighboring photo, this node collects resemblances into this volume. This generated volume consists of noisy resemblances. This node applies a filtering step along X and Y axes which collects local costs which radically decrease the score of large secluded values.
- Finally, the node selects the local minima and substitute the particular plane index with the depth value saved into a depth map volume. This depth map has banding artifacts as it is constructed on the original col- lection of depth values. Moreover, a refining step is applied to compute depth values with sub-pixel accuracy [59].
- All these depth maps could be calculated autonomously as parallel- processing. Later, this node applies a filtering stage to ensure uniformity between various cameras. A compromise is chosen based on both resemblance value and the number of comprehensible cameras to retain inadequately reinforced surfaces without totaling artifacts [59].
- Since this node can take a long time, there is a parameter to allow you to run groups of different cameras as different standalone commands. So if a user has 1000 cameras, then the user could depth process group of cameras with different machines on a farm. Alternatively, running in smaller groups can be useful so that if one machine crashes, the user doesn't have to rerun the whole process [59].

Figure 2.13: Depth Map Estimation [60].

2.3.6.8. Depth Map Filter

- The original depth maps are not wholly consistent; certain depth maps are claimed to see areas that are occluded by other depth maps. This Depth Map filter node isolates these areas and forces depth consistency [59].

2.3.6.9. Meshing

- The objective of this node is to produce a dense geometric surface illustration of the image scene. Firstly, this node fuses every depth map into a global octree where compatible depth values are combined into the octree cells [59].
- Later, this node performs a 3D Delaunay tetrahedralization. Then a complicated voting procedure is implemented to calculate weights on cells and weights on facets linking the cells [59].
- A Graph Cut Max-Flow is applied to cut the volume optimally. This cut signifies the mined mesh surface. This node also filters terrible cells on the surface. Finally, this node applies a Laplacian filtering on the mesh to eliminate local artifacts [59].
- At this stage, the mesh can also be simplified to diminish redundant vertices.

2.3.6.10. Mesh filtering

- In this node, post-processing is performed on the mesh from the meshing node with some refinements. This mesh filtering node performs actions such as smoothing of the mesh, removal of large triangles, keeping the largest mesh but removing all other small mesh [59].
- Some of these operation nodes are not necessarily needed for particular applications so you can custom those parameters as required.

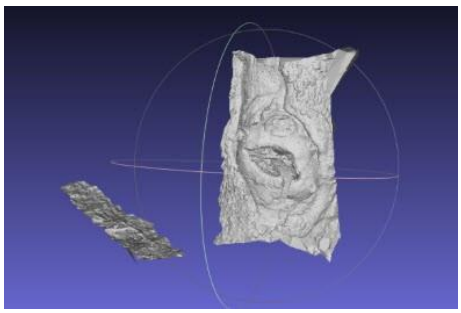


Figure 2.14: Meshing Node [60].

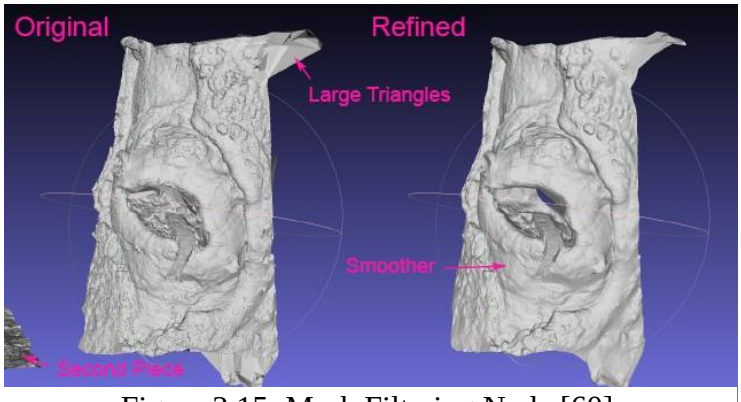


Figure 2.15: Mesh Filtering Node [60].

2.3.7. Occlusion handling

We convert the real reality scene, which we want to support it with virtual objects, into a virtual scene with photogrammetry technology. Occlusion handling is now easier because both the real scene and the object are virtual.

Alignment the real camera coordinates with the virtual camera coordinates. We have obtained the 3D coordinate of each point in the global coordinate system. Theoretically, the correct occlusion relationship between real and virtual object can be obtained by comparing the Z coordinates on them. Compare each Z coordinate of the real scene object Zrealscene (p) with the virtual object Zvirtual (p) in the regions, if Zvirtual (p)< Zrealscene (p), the pixel on the virtual object is drawn on the synthetic image, otherwise, the pixel on the real scene is shown.

2.4.Real-time 3D reconstruction method

2.4.1. Definition

In computer vision and computer graphics, 3D reconstruction is the process of capturing the shape and appearance of real objects. These methods build the 3D model of the real scene and compare the depth of the virtual objects with that of the real scene to handle the occlusion.

To operate in real-time, we need depth sensors such as Kinect sensor, LiDAR sensor. They use a sensor to measure the radiance reflected or emitted by the object's surface to infer its 3D structure through image understanding [61], in order to acquire the depth map.

2.4.2. System overview

As can be seen from Fig. 2.16, our system is divided into two stages: offline stage and online stage.

Offline stage mainly deals with the problem of and 3D reconstruction. With the depth map and RGB information obtained from an RGB-D camera, we reconstruct the real scene using the approach presented in Section 2.4.3.1, which will directly influence the effect of online occlusion handling results.

In online stage, the correct occlusion relationships between real and virtual objects are estimated automatically by comparing the Z coordinates of the virtual objects with that of the real objects, which is introduced in Section 2.4.4 in detail. Subsequently, the synthetic image with correct occlusions is shown in real time.

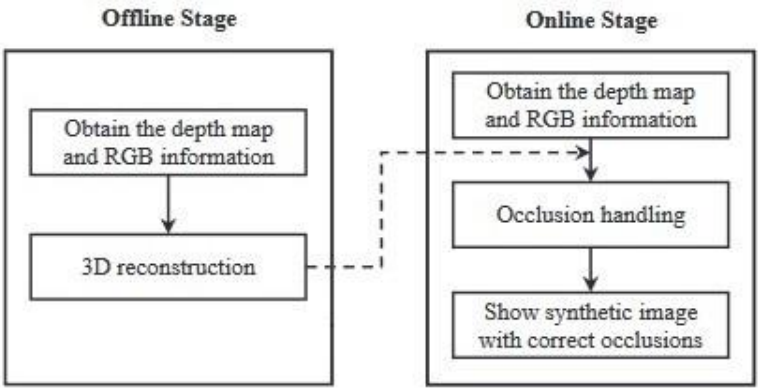


Figure 2.16: Overview of the proposed system.

2.4.3. 3D scene reconstruction

2.4.3.1. Converting depth image into a 3D point cloud

Different with a standard color camera, an RGB-D camera provides RGB image and the corresponding depth value of each pixel. In order to make the next algorithm work (Iterative Closest Point), at each frame k, the depth image must be converted into a 3D point cloud.

As shown in Eq: $V_k(p) = D_k(p)K^{-1}[p, 1]$

we build the 3D vertex $V_k(p) = (x, y, z)$ of each image point p in the camera's coordinate space using the calibration matrix:

$$V_k(p) = D_k(p)K^{-1}[p, 1] \quad (1)$$

Where $D_k(p)$ is the depth value returned by the RGB-D camera, K is the 3 x 3 upper triangular matrix of the infrared camera intrinsic parameters which can be calculated by the camera calibration toolbox of OPENCV, $V_k(p)$ is the 3D vertex corresponding to image point p whose image coordinate is (i , j).

Then the corresponding normal vector for each vertex is computed through cross-product of the neighboring re-projected points as shown in Eq:

$$n_k(p) = \big(V(i+1, j) - V(i,j)\big) \times (V(i, j+1) - V(i,j)) \quad (2)$$

Where $n_k(p)$ is the normal vector of image point p, and then it can be normalized to unit length $n_k(p)/n_k(p)$.

Given the rotation matrix $R = (r_x, r_y, r_z)$ and translation $T = (t_x, t_y, t_z)$ that relate the camera coordinate system to the global coordinate system, the vertex $V_k(p)$ and the corresponding normal $n_k(p)$ at frame k can be converted into global coordinates using Eq :

$$V_k(p) = [R_k \mid T_k]V_k(p) \quad (3)$$
$$N_k(p) = [R_k \mid T_k]n_k(p) \quad (4)$$

2.4.3.2. 3D point clouds alignment based on GPU

Iterative Closest Point (ICP) algorithm is a popular and efficient method for 3D point clouds alignment of two consecutive frames. The basic idea is to find correspondences between the current point cloud at time t and the previous at time t - 1.

Suppose:

- P and Q are two point sets with N point numbers (3 × N matrix);
- $p_i \in$ P and $q_i \in$Q are the same point in different coordinate systems (3 × 1 column vector);
- The rotation matrix and translation matrix between the two coordinate systems are R and S separately.

Then, R and S can be solved by solving optimization problem listed in Eq:

$$\min \sum_{k=1}^{n} ||p_k - (R \times q_k + S)||^2 \qquad (5)$$

The important step is to build the error function of the corresponding points and obtain the minimum error using non-linear optimization approach.

The traditional ICP method has the following disadvantages: first, there are many mismatching points and it is easy to fall into local minimum. second, the iteration speed is slow and it is time consuming especially for large sets of data. To solve these problems, researchers made great efforts to improve the algorithm [63]. The major improvements are list as follows: first, we use the distance from the point to the tangent plane distance (as shown in Figure 2.17(b)) as the error metric instead of the traditional point to point distance (as shown in Figure 2.17(a)).

The traditional point to point error metric is based on the geometrical features of point cloud. Although this method can achieve a precision performance of the point clouds alignment, it takes a long computing time. Because the point to tangent plane metric matches the local geometrical features, the convergence speed is accelerated and the number of iterations is reduced. Moreover, it is less easy to fall into the local extreme. Second, we use the graphic processing unit (GPU) to further accelerate the computing speed and make the algorithm more efficient. GPU is the major processing unit of graphics card. It can exchange data with central processing unit (CPU) in high speed and process large data sets in parallel.

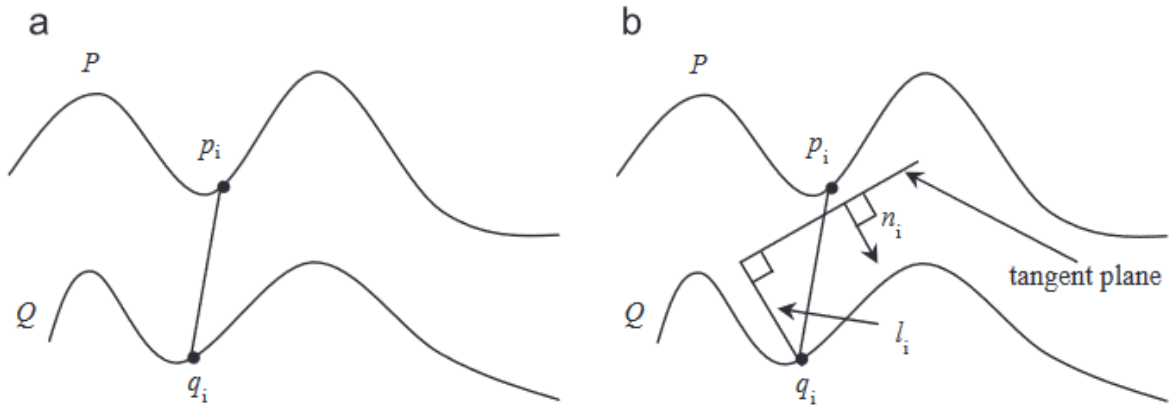


Figure 2.17: (a) Point to point error metric (b) point to tangent plane error metric.

Therefore, most procedures of ICP can be processed by GPU and the computing speed can satisfy the real time requirement. Suppose we need to align the point cloud V_k at frame k to the point cloud V_{k-1} at frame k - 1 (frame k - 1 and frame k are two consecutive image frames), the detailed implementation steps are described as follows:

- Step 1: set the initial iterative matrices. The initial matrices have large impact on the result of iteration.

If the setting is not appropriate, the result is easy to fall into local optimum.

The changes between consecutive image frames are very small, so we can set the initial translation matrix and rotation matrix equal to the previous.

- Step 2: calculate the 3D vertex coordinates and the corresponding normal vectors in parallel using Eq (1) and (2) listed in Section 2.4.3.1
- Step 3: calculate the total error between the point cloud V_k and the point cloud V_{k-1} in parallel using point to tangent plane metric. The equation is listed as follows:

$$\text{error} = \sum_{D_k(p)>0} ||(V_k(p) - V_{k-1}(p)) \times N_{k-1}(p))||^2$$

where $(V_k(p), V_{k-1}(p)$ and $N_{k-1}(p)$are computed using Eq :(3) and (4) presented in Section 2.4.3.1.

- Step 4: Update R_k and T_k and perform Step2 to Step 4 iteratively until the parameter variations of matrices are very small.

Furthermore, to prevent the algorithm running into endless loop, the process should be terminated when error is smaller than a user defined threshold or the number of iterations has reached the user defined maximum value.

2.4.4. Occlusion handling

In the offline stage, we have obtained the 3D coordinate of each point in the global coordinate system. Theoretically, the correct occlusion relationship between real and virtual object can be obtained by comparing the Z coordinates on them. The superposed virtual object is usually very complex and irregular, so it is very difficult to get the coordinate of each point on the virtual object.

To deal with this problem, we use the following method to acquire correct occlusion relationships in online stage:

- First, render the 3D virtual object to the 2D image plane using rotation matrix R' and translation matrix T'. We suppose that s represents the region in the image where 3D virtual object rendered.
- Second, get the corresponding depth value for each pixel p where p ϵ s. Then we can use back projection method to map the image coordinates p into the object coordinates P ϵ S in current camera coordinate system.
- Third, align the current 3D cloud point to the previous 3Dmodel using the method demonstrated in Section 2.4.3. Therefore, the transformation relations between the current frame and the previous one is known. Then the 3D coordinate of each point in current camera coordinate system can be calculated.
- Fourth, compare each Z coordinate of the real object Zreal(p) with the virtual object Zvirtual (p) in the regions, if Zvirtual(p)< Zreal(p), the pixel on the virtual object is drawn on the synthetic image, otherwise, the pixel on the real scene is shown.

The online occlusion handling procedure can be operated in real time for two reasons: first, we just compare the Z coordinate of each pixel in the region rendered by the virtual object. This will largely reduce the processing time and meet the real time requirement. Second, in the 3D

point cloud alignment stage, we use GPU to accelerate the computation speed and make the algorithm more efficient.

2.5. Conclusion

Methods handling occlusions in augmented reality are tricky. Despite the multiplicity of innovative technologies. Nevertheless, we examined the best suggested approaches to occlusion handling.

In this chapter, we have detailed the techniques Photogrammetry for 3D reconstruction method and Real-time 3D reconstruction method and how to occlusion handling for each method.

Based on our study of this area, we found that both techniques have advantages and disadvantages. This prompted us to propose our contribution, which will be described in the following chapter.

Chapter 3

3. Handling occlusions based on 3D reconstruction in augmented reality

3.1. Introduction

In this chapter we will explain the steps and modules composing the two systems (Photogrammetry for 3D reconstruction method, Real-time 3D reconstruction method), where we present the design for every system, we have by starting with its Global design then its Detailed design by explaining the different elements of the system and specifying their operation.

3.2. Photogrammetry for 3D reconstruction method

3.2.1. Methodology

In our system, we'll handling occlusions on a fixed scene, take photos at the scene with technique photogrammetry for 3D reconstruction method. This technology allows us to convert multiple images of a real scene into a virtual scene (realscene.obj) file formats, and when adding a virtual object (virtual.obj) to the real scene. We show the result of occlusion handling between this object (virtual.obj) and the virtual scene (realscene.obj).

3.2.2. Global system design

Generally, our system handling occlusions in augmented reality will be following a certain step, as we represent the global architecture in figure 3.1.

As we noticed in the previous system architecture, three phases are part of system:

The Front-end:(User Interface)

Which is the important phase for the user where the interaction on the phone screen or any computer connected to a camera, also the results displaying after getting them from the Back-end phase.

The Back-end:

The most important phase in the system, during this process, we will receive 3D virtual model for real scene, we use the model tracking method to Projection the virtual scene onto the real scene. we adding virtual object and occlusion handling and get the correct result.

The Photogrammetry:

The first phase in the system, during this process, we'll scan all of the real scene by taking multiple images from different angles. we process the images to form a virtual model of the real scene.

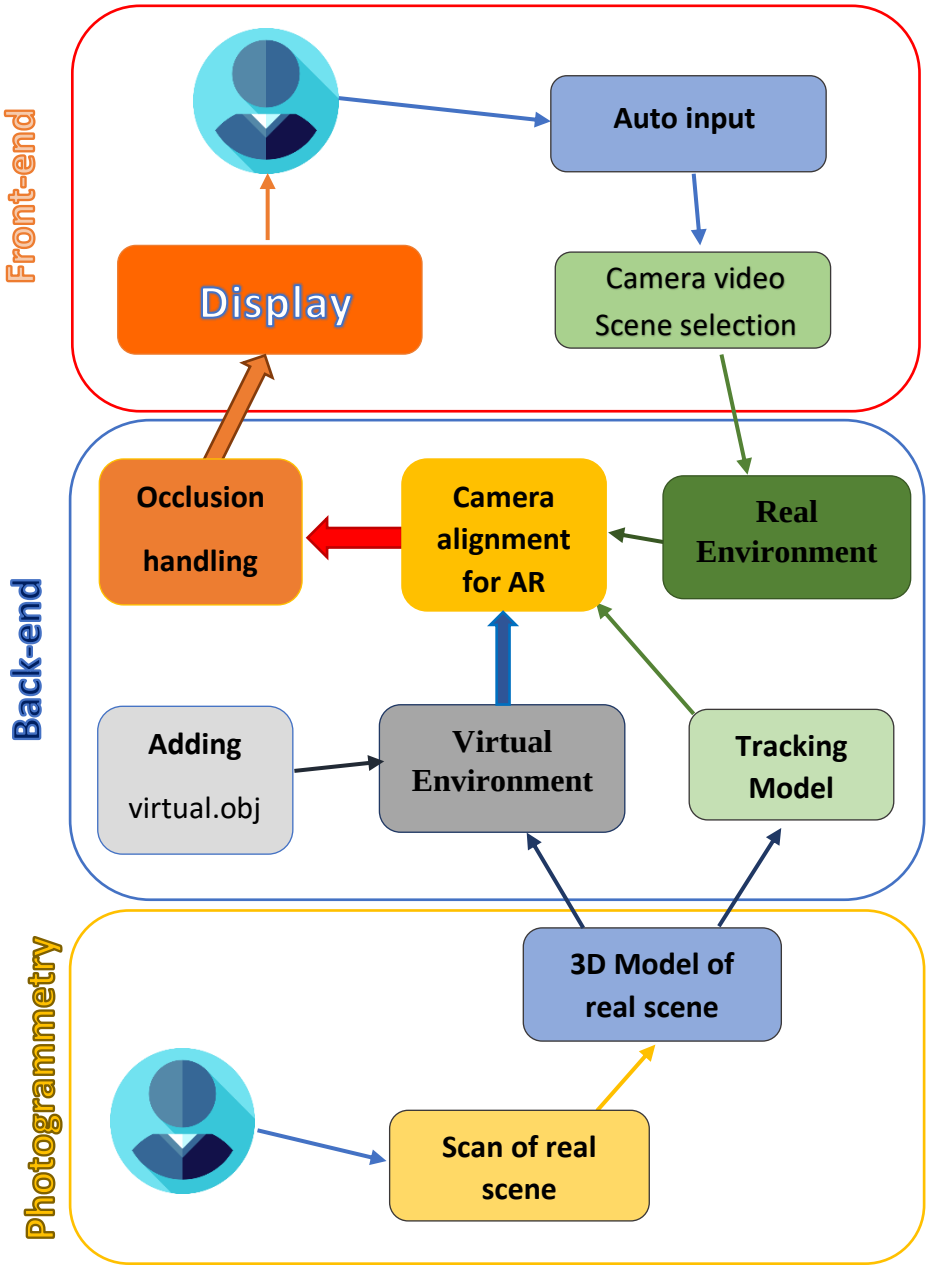


Figure 3.1: Global system Design.

3.2.3. Detailed system design

3.2.3.1. Photogrammetry

We will divide this phase into two basic steps:

A. Scan of real scene

We choose a real scene to apply the experiment to occlusion handling. This scene contains objects small in size to facilitate studying. We prepare a place to take images. So that we place a white cloth under the scene of the objects in the middle of a room with a stable and calm background to avoid detecting random points, using the approach presented in Section 2.3.4.2. The scene should be illuminated properly with the light used to avoid light reflections from the

object and dark shadows as explained in Section 2.3.4.1. Pictures are taken around the object clockwise and combine varying heights and viewing angles for each position. The consecutive images should have a difference of 20 to 30 degrees. Camera and Lens Quality The accuracy of the cameras is essential in determining the accurate location of targets and markers. We will be using a high-quality camera of 4608 x 3456P, 16M Pixels to take images.

B. 3d Model of real scene

We put the real scene photos in the program Meshroom. We are waiting for the program to obtain a 3d model. So that the 3d model is a virtual object of the real scene in high percentage.

3.2.3.2. Back-end

The Back-end phase it's the main body of the system where we gonna pass thought important steps described in following:

A. Tracking Model

We prepare the object (realscene.obj) for tracing.

So that we exploit software that is able to create a model with features to know the location of the similar shape in reality. 3D Model Tracking technology is covered in detail in section 1.2.1. We create trace files with a 360-degree angle with a (realscene.obj) in the middle to fully track the form.

B. adding virtual object

We choose a virtual object that matches the scene. So that the object in *.obj file formats. This object will be the foreign object in the real scene. To which we will apply the Occlusion handling to take its rightful place in the scene.

C. Virtual Environment (VE)

We create a virtual environment similar to the real scene in all the details: the number of objects and sizes. After we performed the photogrammetry of the study scene and create a 3D model to the real scene. We rely on this model to create a virtual environment for the study scene. We are adding a virtual object within this environment to study the occlusion.

The virtual environment has a virtual camera that moves in all directions to photograph the complete environment.

D. Real Environment (RE)

We receive real-time video camera recording of the real scene. In every frame of the video, we receive it in the form of a 2D image. We set the 2D image as a static background for the real camera. It shows us the real world in a virtual world.

E. Camera alignment for AR

Augmented reality (AR) is the merging of the real environment with the virtual environment.

The real camera it's a fixed coordinate.

$$\boldsymbol{C_{Real}(x, y, z)} = (0,0,0).$$

The virtual camera it's a fixed coordinate.

$$\boldsymbol{C_{Virtual}(x, y, z)} = (0,0,0).$$

The model tracking information results help us determine virtual camera coordinates from the real environment.

We search for gestures for the object (realscene.obj) in the video on every frame and we apply the projection. We project a 3D model onto the 2D frame as in figure 1.2. With the help of the coordinates transformation calculation software (Vuforia), we pass the coordinates result to the virtual camera.

$$\boldsymbol{C_{Virtual}(x, y, z) = (0, 0, 0) * Transformation\_Vuforia}\ .$$

We align the real camera coordinates with the virtual camera coordinates and watch the virtual scene with the real scene behind it.

$$\boldsymbol{C_{Real}(x, y, z) = C_{Virtual}(x, y, z)}.$$

F. Occlusion handling

We create a virtual reality environment with a virtual camera.

By combining the coordinates of the real camera with the virtual camera, augmented reality is produced for us. In a virtual environment, we compare the Z coordinates of the two objects (realscene.obj, virtual.obj). Using the method demonstrated in Section 2.3.7.

3.2.3.3. Front-end

In our front-end system, the main part for the user-backend interaction, so that it sends the real scene information by the video camera, the result is received on the display screen.

3.3.Real-time 3D reconstruction method

3.3.1. Methodology

In our system, we will be using depth sensors, and processing its dataset to create a virtual environment for real scene (realscene.obj) with technique Real-time 3D reconstruction method. This technology allows us to handling occlusions in real time for real scenes. when adding a virtual object (virtual.obj) to the real scene. We show the result of occlusion handling between this object (virtual.obj) and the virtual scene (realscene.obj).

3.3.2. Global system design

Generally, our system handling occlusions in augmented reality will be following a certain step, as we represent the global architecture in figure 3.2.

As we noticed in the previous system architecture, two phases are part of system:

The front-end:(User Interface)

which is the important phase for the user where the interaction and the requests of files and dataset are done, also the displaying and the storage of dataset and results after getting them from the Back-end phase.

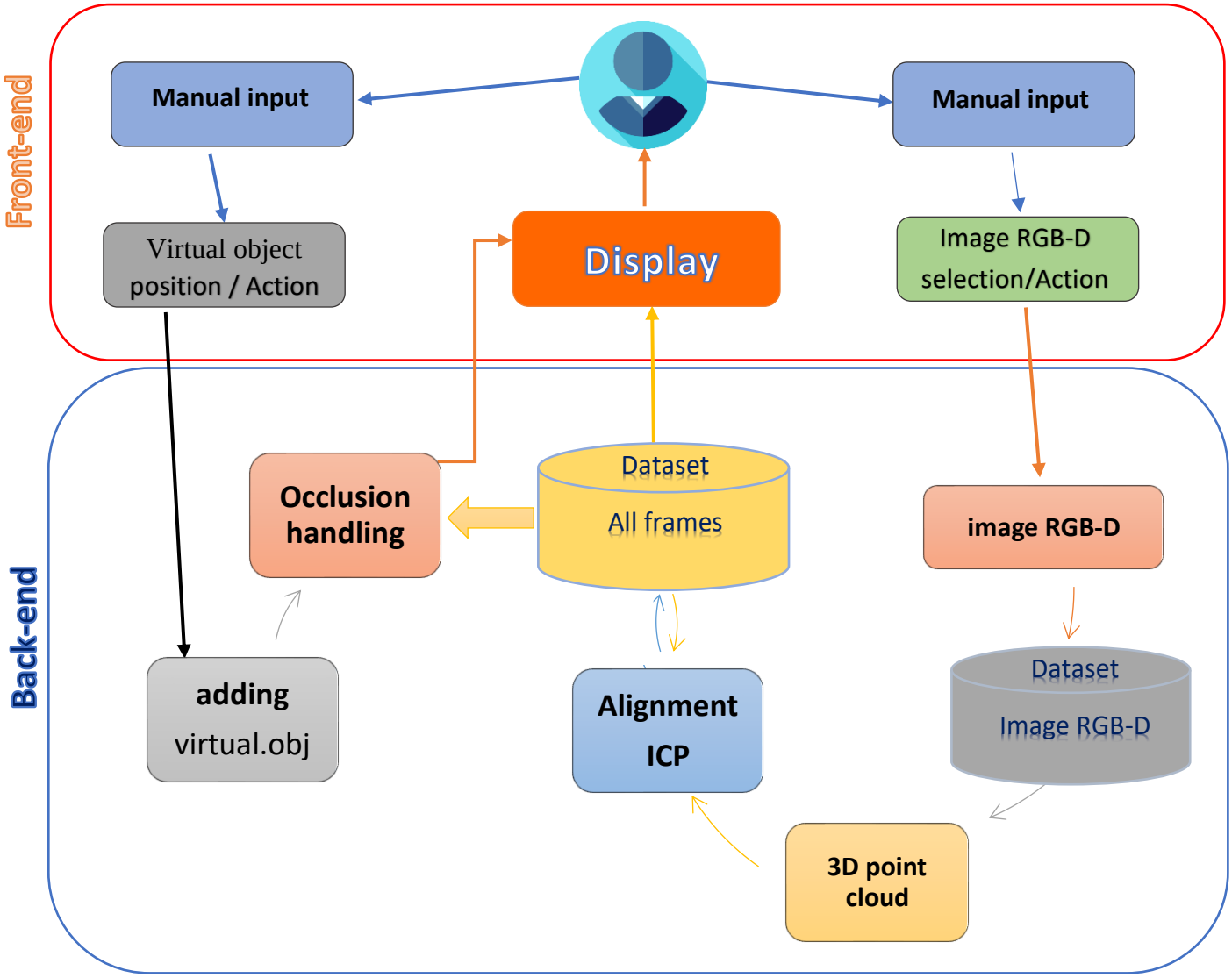


Figure 3.2: Global system Design.

The Back-end:

The most important phase in the system, during this process, we will receive dataset from the depth sensors, analyze and process it to create a virtual environment model for the real scene. we use this form to occlusion handling and get the correct result.

3.3.3. Detailed system design

3.3.3.1. Back-end

The Back-end phase it's the main body of the system where we gonna pass thought important steps described in following:

1. Receive image RGB-D.
2. Processing the image RGB-D.

3. Create 3D point clouds.
4. Alignment of two consecutive frames.
5. Save datasets of all frames.
6. adding virtual object.
7. Occlusion handling.

the Back-end phase in detail:

A. Receive image RGB-D

There are RGB-D camera sensors devices, among these devices we will rely on the depth sensors of camera Kinect see fig 3.1. We import images RGB-D as digital matrices.

Figure 3.3: the depth sensors of camera Kinect.

B. Processing the image RGB-D

After receiving matrices for images, we extract the colors values of each pixel from the RGB images. So that each pixel contains 3 color values (R, G, B). we convert the Depth image to space HSV (hue, saturation, value).

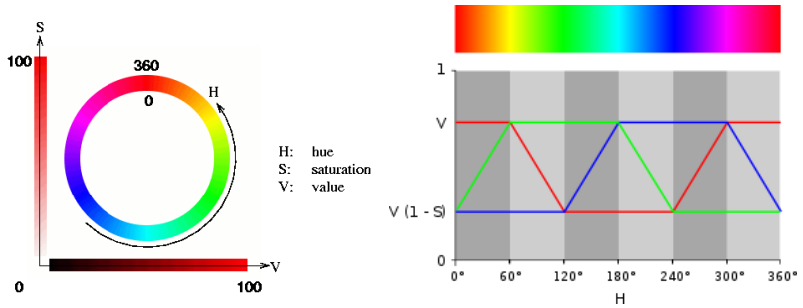


Figure 3.4: A graphical explanation of RGB coordinates given values for HSV.

C. Create 3D point clouds

We transform the dimension matrices to 3D point clouds.

we build the 3D vertex $\boldsymbol{V_k(p) = (x, y, z)}$ of each image point p.

$$\boldsymbol{V_0(p_{(i,j)}) = \left(\sum_{i=1}^{n} i \ , \sum_{j=1}^{m} j \ , 0 \right)}$$

Using the previous equation, we created an image RGB for real scene into a virtual scene.

and adding depth to the virtual scene:

$$V_0(p_{(i,j)}) = \left(\sum_{i=1}^{n} i \ , \sum_{j=1}^{m} j \ , \mathbf{depth\_H}_{(i,j)} \right)$$

Using the previous equation, we created an image RGB-D for real scene into a virtual scene.

Section 2.3.3.1 describes the system overview in detail.

D. Alignment of two consecutive frames

We align the current 3D point clouds with the final previous frame of the scene.

We choose current frames $F_{current}$ and F_{final} and apply an ICP algorithm to extract the rotation matrix (R) and translation matrix (T) of the second frame as we saw in chapter 2 section 2.3.3.2. We apply the two matrices R and T to the coordinates of the scene $F_{current}$ by aligning it with the scene F_{final}.

E. Save datasets of all frames

We combine the scenes in one scene. We save the datasets in a file that can be used later in occlusion handling. This file contains all the scene information.

F. adding virtual object

We choose a virtual object that matches the scene. So that the object in *.obj file formats. This object will be the foreign object in the real scene. To which we will apply the Occlusion handling to take its rightful place in the scene.

G. Occlusion handling

Now we can handle occlusions we have two components of the same type, virtual real scene and virtual object. You can place the object in its correct position, show only the part that is visible to the user, and hide the covered part. By comparing the depth coordinates for each point of the object. using the method demonstrated in Section 2.4.4.

3.3.3.2. Front-end

In our front-end system, the main part for the user-backend interaction, Giving the ability of exchanging data between user-backend, we gonna make simple access design for each part, and each request will interact with the backend related to. The Figure 3.8 will illustrate the whole mechanism.

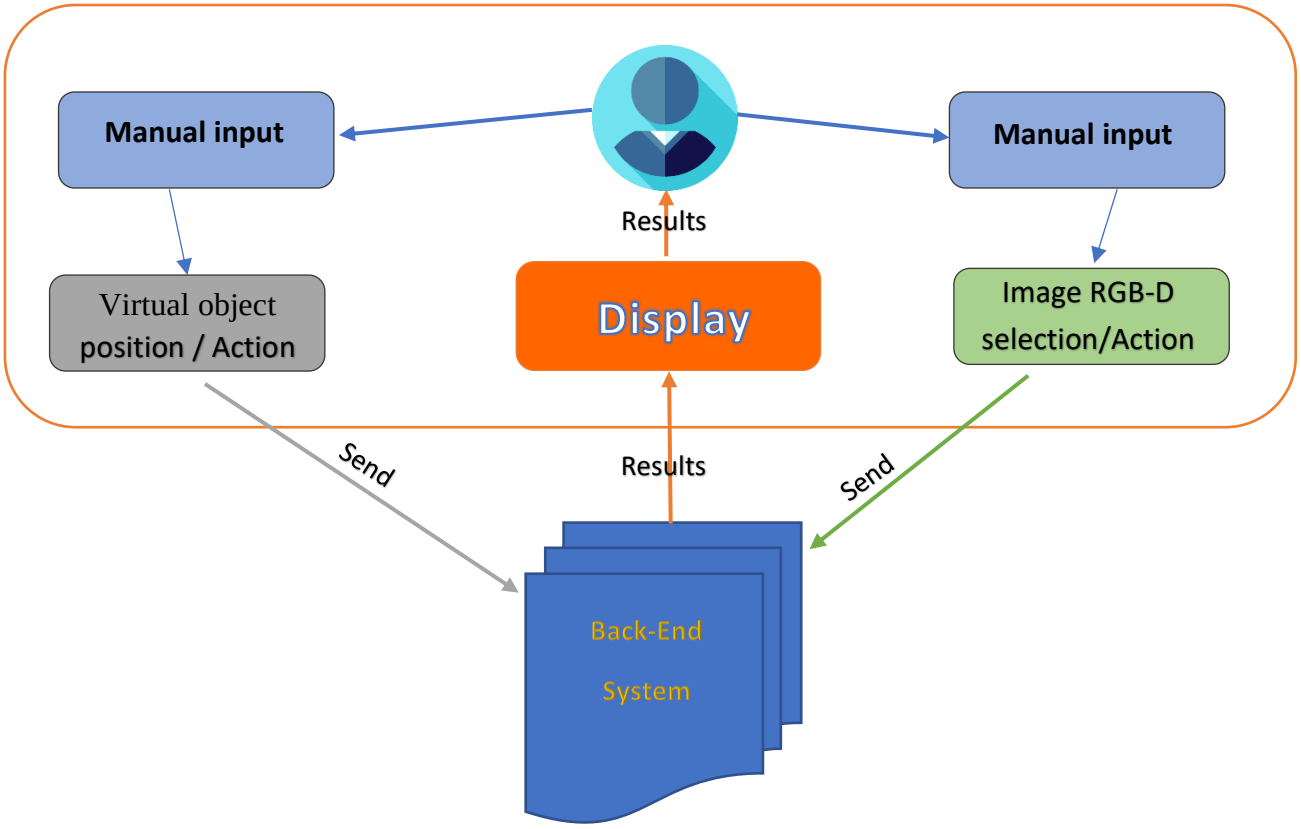


Figure 3.5: Front-End System.

3.4.Conclusion

In this chapter, we have presented our proposed method, where we have presented the design of our the two systems (Photogrammetry for 3D reconstruction method, Real-time 3D reconstruction method), and we have detailed the steps that we have passed to arrive at the two systems, and detailed the modules used in Photogrammetry for 3D reconstruction method and the three essential phases (Front-end, Back-end and Photogrammetry), and the modules used in Real-time 3D reconstruction method and the two essential phases (Front-end, Back-end).

In the next chapter, we will expose the implementation of the two systems.

Chapter 4

4. Implementation and results

4.1.Introduction

In this chapter, we will present the working environment, the programming language, and the tools we used to build the two systems (Photogrammetry for 3D reconstruction method, Real-time 3D reconstruction method). Subsequently we will explain all the experiments that we have applied to the proposed methods and the results obtained.

4.2.Implementation

4.2.1. Photogrammetry for 3D reconstruction method

4.2.1.1. Environments and developing tools

To Develop our system, we are going to use different environments and tools for the backend programming language including API's, libraries, SDK, and for the front-end also we gonna use programming language, working on many editors, IDE etc....

Meshroom

Meshroom is a free, open-source 3D Reconstruction Software based on the AliceVision Photogrammetric Computer Vision framework. This is providing 3D reconstruction algorithms. The project is a result of collaborations between academia and industry to provide cutting-edge algorithms with the robustness and the quality required for production usage.

CUDA

CUDA is a parallel computing platform and application programming interface model created by Nvidia. It allows software developers and software engineers to use a CUDA-enabled graphics processing unit for general purpose processing — an approach termed GPGPU.

MeshLab

MeshLab the open source system for processing and editing 3D triangular meshes. It provides a set of tools for editing, cleaning, healing, inspecting, rendering, texturing and converting meshes. It offers features for processing raw data produced by 3D digitization tools/devices and for preparing models for virtual reality and augmented reality.

Blender

Blender is a free and open-source 3D computer graphics software toolset used for creating animated films, visual effects, art, 3D printed models, motion graphics, interactive 3D applications, virtual reality and computer games. Blender's features include 3D modeling, UV unwrapping, texturing, raster graphics editing, rigging and skinning, fluid and smoke simulation, particle simulation, soft body simulation, sculpting, animating, match moving, rendering, motion graphics, video editing, and compositing.

Unity

Unity is a cross-platform game engine with a built-in Integrated Development Environment (IDE) by Unity Technologies. The engine can be used to create three-dimensional, two-dimensional, virtual reality, and augmented reality games, as well as simulations and other experiences.

Visual Studio

Visual Studio is an Integrated Development Environment (IDE) developed by Microsoft to develop GUI (Graphical User Interface), console, desktop applications, web apps, mobile apps, cloud, and web services, etc. With the help of this IDE, you can create managed code as well as native code. It uses the various platforms of Microsoft software development software like Windows store, Microsoft Silverlight, and Windows API, etc. It is not a language-specific IDE as you can use this to write code in C#, C++, VB (Visual Basic), Python, JavaScript, and many more languages. It provides support for 36 different programming languages. It is available for Windows as well as for macOS.

C#

C# (C Sharp) is widely used for developing desktop applications, web applications and web services. It is used in creating applications of Microsoft at a large scale. C# is also used in game development in Unity.

Vuforia

Vuforia is an augmented reality software development kit (SDK) for mobile devices that enables the creation of augmented reality applications. It uses computer vision technology to recognize and track planar images and 3D objects in real time.

Vuforia MTG

The Model Target Generator (MTG) converts an existing 3D model into a Vuforia Engine database that can be used by the Vuforia Engine for Model Target tracking. This tool enables you to confirm whether the features of your model will be usable, to set up the initial snapping position, and then export the final database.

Android Studio

Android Studio is the official Integrated Development Environment (IDE) for Android app development, based on IntelliJ IDEA. You can build apps for Android phones, tablets, Android Wear, Android TV, and Android Auto. Structured code modules allow you to divide your project into units of functionality that you can independently build, test, and debug.

Android SDK

The Android SDK (software development kit) is a set of development tools used to develop applications for Android platform. The Android SDK includes sample projects with source code, development tools, an emulator, and required libraries to build Android applications.

4.2.1.2. Developing the Photogrammetry

In this part of the system we are going to develop the photogrammetry system, where we shoot the real scene, we chose it for the study. And create a 3D model of the scene. We use the phone camera to take multiple images of high quality and Meshroom program to convert multiple images into a 3D model.

4.2.1.2.1. Scan of the real scene

In this part, we prepare the real scene for photoshoot. Our real scene contains 3 objects. They are small kitchen objects (Refrigerator, Electric oven, Cooking pot) see in fig 4.1.

Figure 4.1: the real scene is small kitchen objects (Refrigerator, Cooking pot, Electric oven).

To capture multiple images for the study scene, we use the phone's camera Huawei Y9 (2019). Huawei Y9 camera specification is mentioned in the following Table 4.1. The ambiance was well lit with diffused ceiling light so that noises can be removed from the photogrammetric process. The camera was set to capture High Dynamic Range (HDR) images, These HDR files generate image of the size in between 2-4 MB. However, here is the catch, for photogrammetry pipeline only JPEG images are allowed. If images are captured in JPEG format.

4.2.1.2.2. 3D Model of the real scene

We use Meshroom program to convert multiple images of the Real scene into 3d model. Where the user interface of the program on the left side provides a place to place the captured images, shown in the figure 4.2. Click button start to start the process of converting images into 3d model, by following the photogrammetry algorithm of Meshroom which were explained in Section 2.3.6.

4.2.1.3. Developing the Back-end

In this part of the system, we are going to develop the backend system where receives information from two systems (Photogrammetry, Front-end) and processes it to create a real environment and a virtual environment, using Unity IDE. We will use other software to help us in our work to occlusion handling.

Table 1.1: Phone Huawei Y9 (2019) Camera Specifications.

System information	Specifications
Type	Mobile phones
Indicative Price	USD 219.90
Resolution	4608 x 3456
color depth	24
Type of Sensor	CMOS
Sensor photo detectors	16 Megapixels (**MP**)
shooting modes	Continuos Shooting, High Dynamic Range mode (**HDR**)
Focal length multiplier	F/1.8
Frame rate	1/25 **fps**

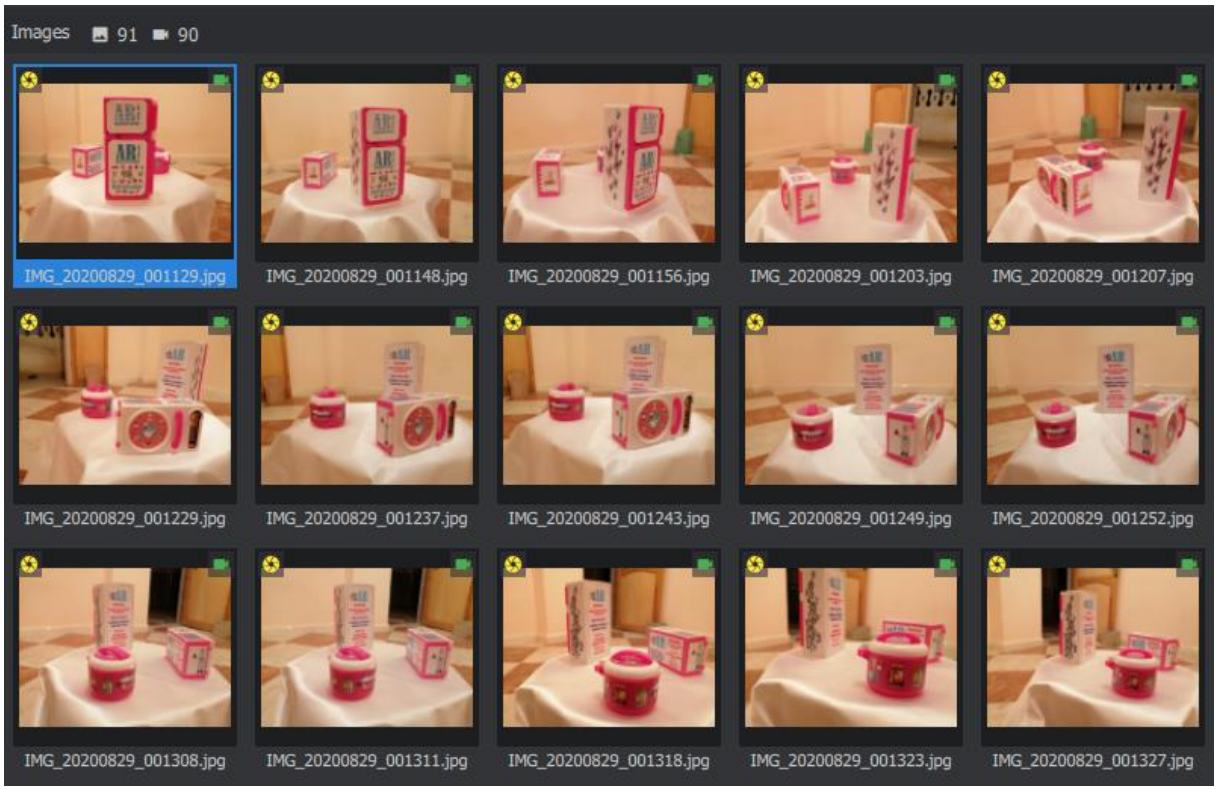


Figure 4.2: Meshroom on the left side provides a place to place the captured images.

4.2.1.3.1. Tracking Model

We use a Vuforia MTG program to create features for a 3D model of the real scene to facilitate the positioning of objects and their distance from the camera.

We are filtering the 3D model With MeshLab program into individual objects (Refrigerator, Cooking pot, Electric oven), see fig 4.3. We fixing the axes for the objects with Blender program, see fig 4.4. We upload each object to Vuforia MTG program and create special features for each object. And we define an angle of 360 from all sides. We follow the object in high detail, see fig 4.5.

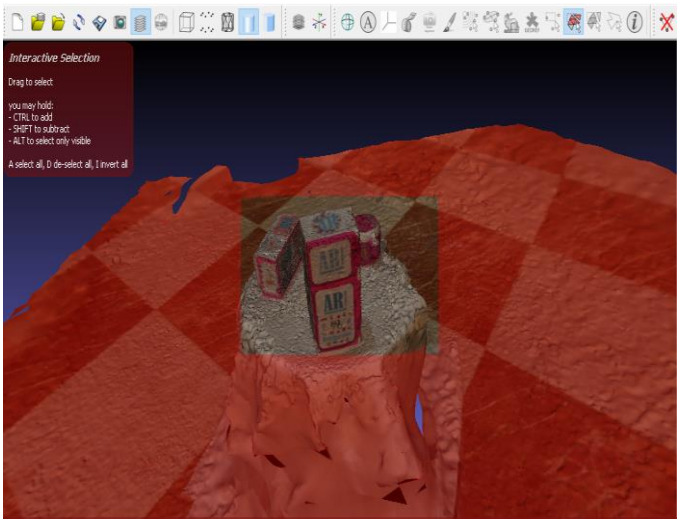


Figure 4.3: filtering the 3D model With MeshLab program.

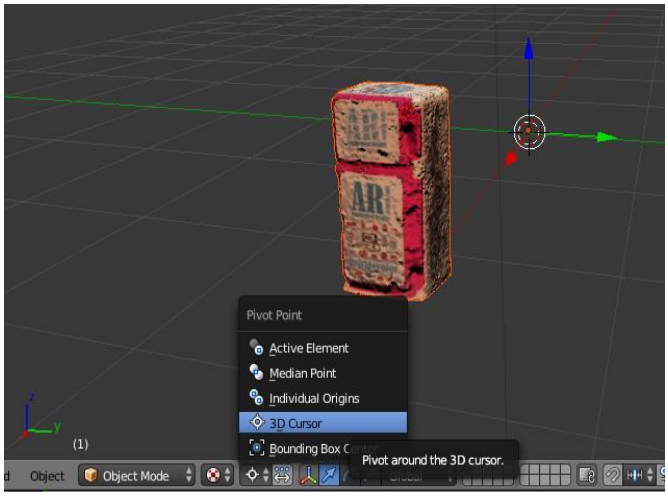


Figure 4.4: fixing the axes for the objects with Blender program.

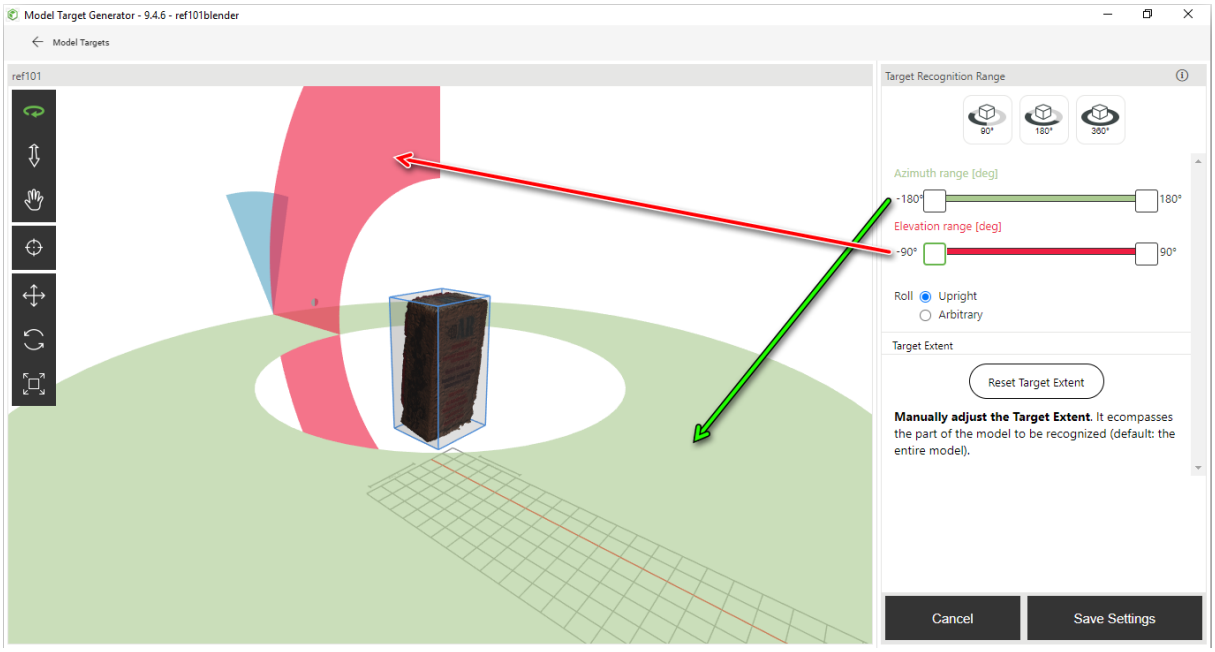


Figure 4.5: create special features for refrigerator object.

4.2.1.3.2. adding virtual object

We choose a virtual object on which we will study the occlusion with the real scene. And we choose file stove.obj see fig 4.6 to add it to the virtual environment.

Figure 4.6: file stove.obj.

4.2.1.3.3. Virtual Environment (VE)

We create a virtual scene similar to reality based on the 3D model of the real scene. We add the Stove object (stove.obj) to the virtual environment. And make its coordinates fixed between objects see fig 4.7. And we make the coordinates of the virtual environment camera see fig 4.8:

$$\boldsymbol{C_{Virtual}}(\boldsymbol{x}, \boldsymbol{y}, \boldsymbol{z}) = (0,0,0).$$

Figure 4.7: coordinates stove.obj fixed between objects.

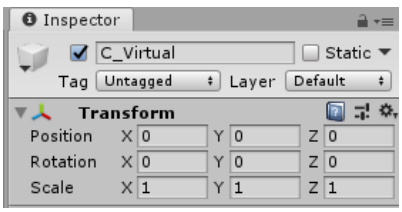


Figure 4.8: the coordinates of the virtual environment camera.

4.2.1.3.4. Real Environment (RE)

We receive real-time video camera recording of the real scene. we receive it in the form of a 2D image. We set the 2D image as a static background for the real camera see fig 4.9. It shows us the real world in a virtual world. And we make the coordinates of the real environment camera see fig 4.9:

$$C_{Real}(x, y, z) = (0,0,0).$$

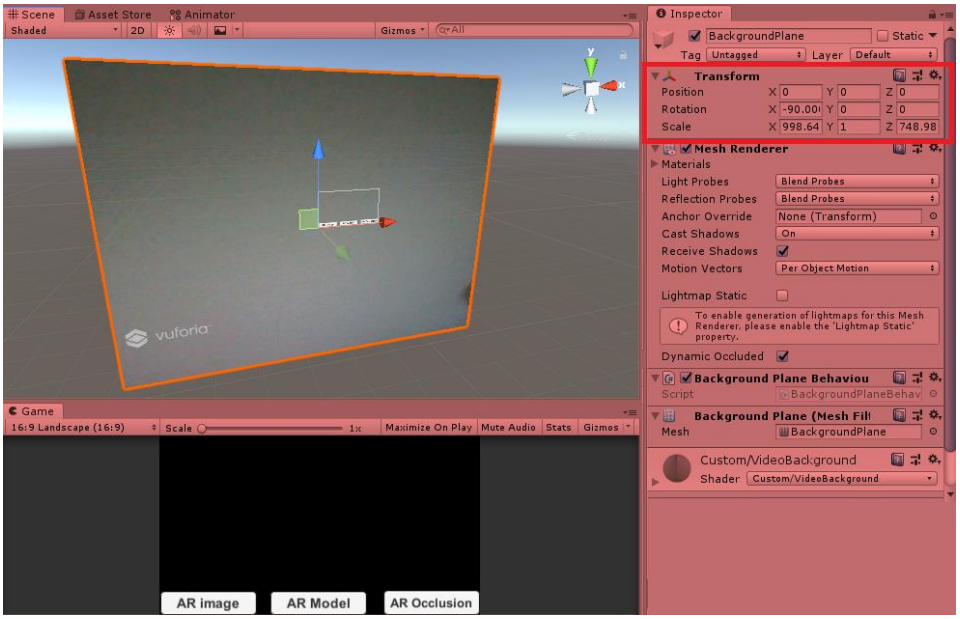


Figure 4.9: 2D image as a static background and coordinates camera in the real environment.

4.2.1.3.5. Camera alignment for AR

We align the real and the virtual cameras together to configure the alignment camera (ARCamera). We create this camera from Vuforia Engine package in Unity see fig 4.10. The ARCamera is aligned with the two cameras to merge the real and virtual environment in one camera. The ARCamera coordinates move according to the tracking model.

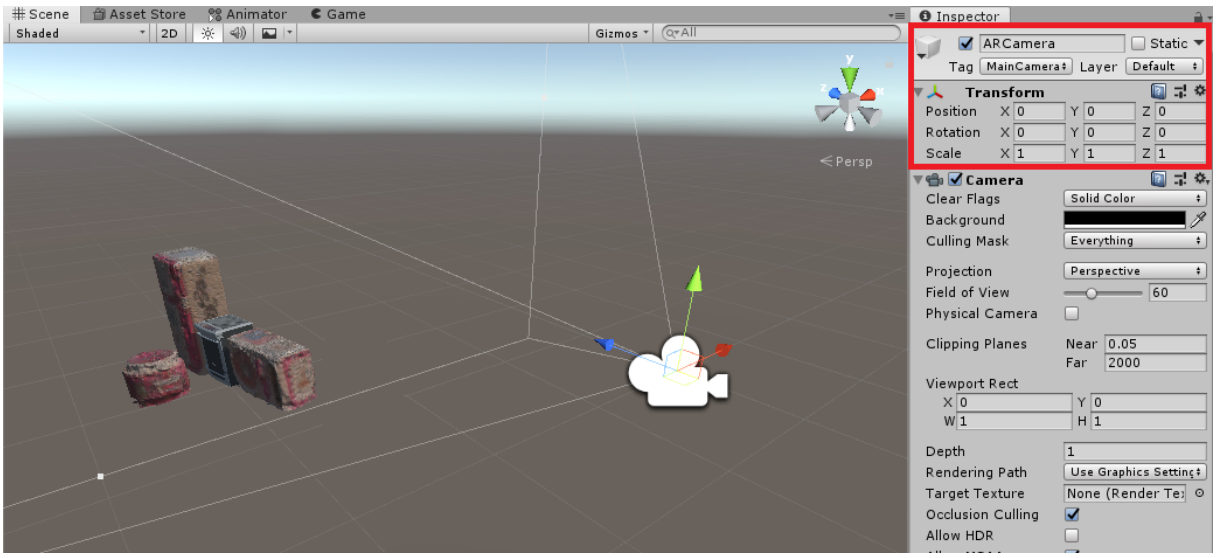


Figure 4.10: The ARCamera coordinates in Unity.

G. Occlusion handling

We dress the virtual object for a real scene Shader DepthMask see fig 4.11. This allows the material to show which objects in front of the object and those behind it disappear from the view. It hides the part between the object and the camera's field of view.

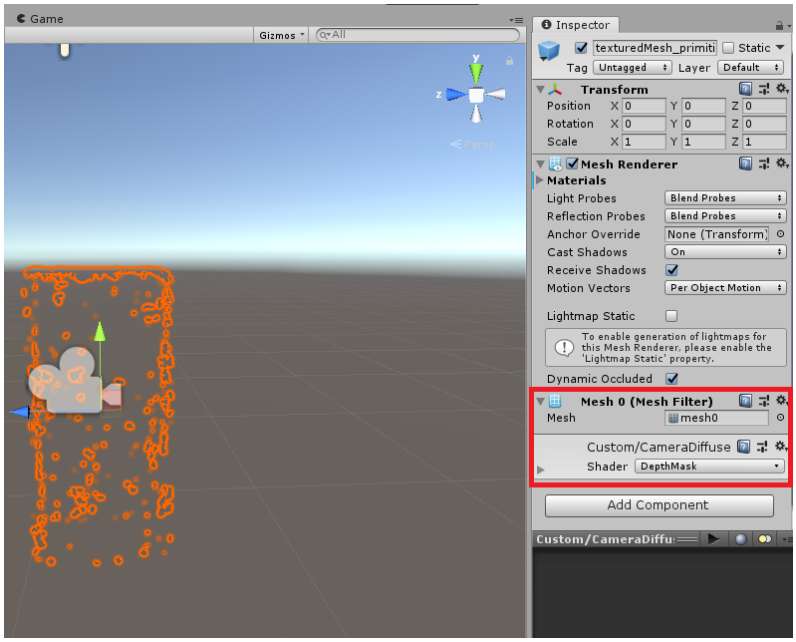


Figure 4.11: dress the virtual object for Shader DepthMask.

4.2.1.4. Developing the Front-end

In the front-end part of the system, we gonna create a multi-pule user interface pages, we want to create a simple design that can every user with different gender and ages to use it, with some simple colors, and structure. The figure 4.12 shows the architecture of the Front-end pages in program.

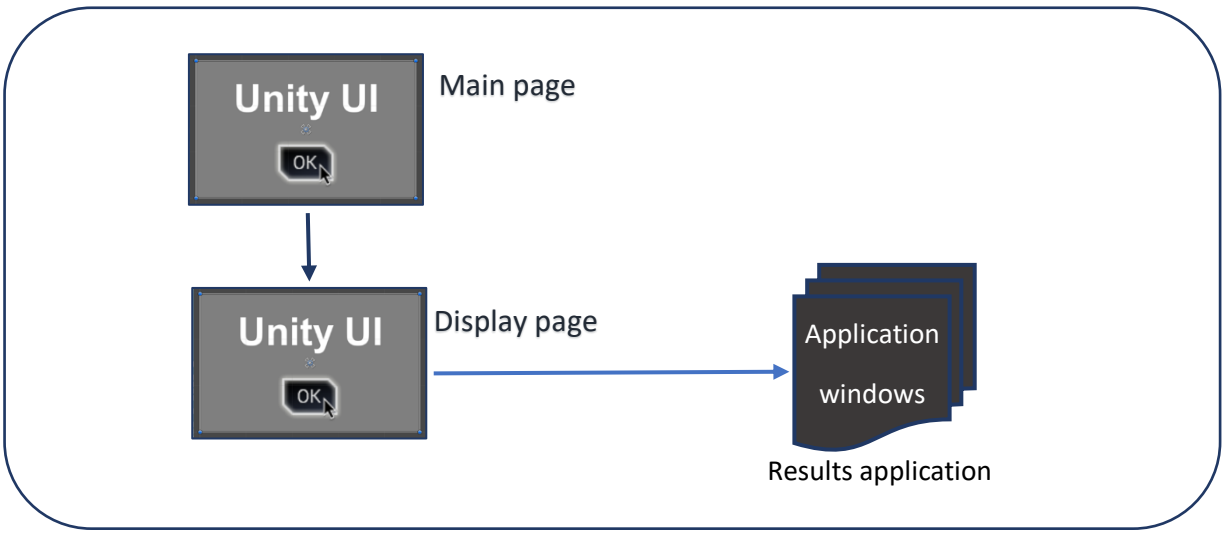


Figure 4.12: shows the architecture of the front-end pages in Unity.

Working on Unity for the content and the structure of the things that we want to put in out pages, With the use of the Unity UI which give us a big advantage of designing and styling our page for better looking and well organizing, He also makes our pages moving and working giving some benefits on animations, button functions, pages function, Processing, display, etc..., the figures(4.13 , 4.14) below shows the content of each created Unity UI page.

Figure 4.13: Main page.

Figure 4.14: Display page.

4.2.2. Real-time 3D reconstruction method

4.2.2.1. Environments and developing tools

To Develop our system, we are going to use different environments and tools for the backend programming language, libraries, toolbox, and for the front-end also we gonna use programming language, working on many editors, IDE etc... .

MATLAB

MATLAB is a high-performance language for technical computing. It integrates computation, visualization, and programming in an easy-to-use environment where problems and solutions are expressed in familiar mathematical notation. Typical uses include :

- Math and computation.
- Algorithm development.
- Modeling, simulation, and prototyping.
- Data analysis, exploration, and visualization.
- Scientific and engineering graphics.
- Application development, including Graphical User Interface building.

MATLAB GUI

Graphical user interfaces (GUIs), also known as apps, provide point-and-click control of your software applications, eliminating the need for others to learn a language or type commands in order to run the application. You can share apps both for use within MATLAB and also as standalone desktop or web apps.

MATLAB Image Processing Toolbox

Processing Toolbox™ provides a comprehensive set of reference-standard algorithms and workflow apps for image processing, analysis, visualization, and algorithm development. ... You can interactively segment image data, compare image registration techniques, and batch-process large data sets.

MATLAB Computer Vision Toolbox

Computer Vision Toolbox™ provides algorithms, functions, and apps for designing and testing computer vision, 3D vision, and video processing systems. You can perform object detection and tracking, as well as feature detection, extraction, and matching. For 3D vision, the toolbox supports single, stereo, and fisheye camera calibration; stereo vision; 3D reconstruction; and lidar and 3D point cloud processing. Computer vision apps automate ground truth labeling and camera calibration workflows.

Visual Studio

Visual Studio is an Integrated Development Environment (IDE) developed by Microsoft to develop GUI (Graphical User Interface), console, desktop applications, web apps, mobile apps, cloud, and web services, etc. With the help of this IDE, you can create managed code as well as native code. It uses the various platforms of Microsoft software development software like Windows store, Microsoft Silverlight, and Windows API, etc. It is not a language-specific IDE as you can use this to write code in C#, C++, VB (Visual Basic), Python, JavaScript, and many more languages. It provides support for 36 different programming languages. It is available for Windows as well as for macOS.

C++

C++ is a powerful general-purpose programming language. It can be used to develop operating systems, browsers, games, and so on. C++ supports different ways of programming

like procedural, object-oriented, functional, and so on. This makes C++ powerful as well as flexible.

OpenGL

OpenGL® (Open Graphics Library) is a cross-platform library for interfacing with programmable GPUs for the purpose of rendering real-time 3d graphics. Its use is common in games, CAD, content creation, energy, entertainment, game development, manufacturing, medical, and virtual reality.

4.2.2.2. Developing the Back-end

In this part of the system we are going to develop the backend system where receives information from depth sensors and processes it to create a virtual environment for a realistic scene, using Matlab language with Matlab GUI interface and Toolbox and libraries that processing and analyze images. And use Visual Studio IDE C++ and OpenGL graphics library to occlusion handling.

4.2.2.2.1. Receive image RGB-D

The Matlab IDE provides us with an image analysis library MATLAB Image Processing Toolbox, which helps us to receive images in digital matrices.

```
% load image RGB
path= imgetfile();
img_rgb= imread(path);
% load image depth
path2= imgetfile();
img_depth= imread(path2);
```

Listing 4.1: load image in Matlab.

4.2.2.2.2. Processing the image RGB-D

After receiving the information from the depth sensors of camera Kinect we analyze in MATLAB:

— **First**, Extract the colors values of each pixel from the (a)RGB image in figure 4.15.

each pixel contains 3 color values (R,G,B) .

```
R_color = img_rgb(:,:,1); % R (0-255).
G_color = img_rgb(:,:,2); % G (0-255).
B_color = img_rgb(:,:,3); % B (0-255).
```

Listing 4.2: analyze image RGB.

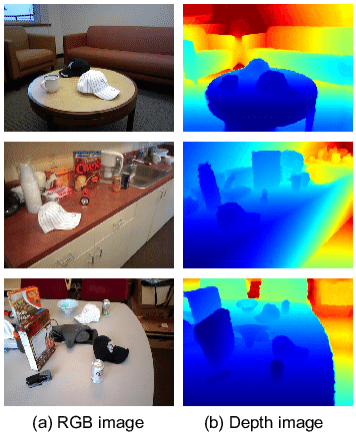


Figure 4.15. An example of with image RGB-D Captured for a camera Kinect.

All captured image in figure 4.15 is of size (170 x 128) pixel.

— **Second**, Convert the (b)Depth image in figure 7 to space HSV (hue, saturation, value).

```
img_hsv = rgb2hsv(img_depth);
depth_H = img_hsv(:,:,1); // depth_H (0-359).
```

Listing 4.3: analyze depth image.

4.2.2.2.3. Create 3D point clouds

MATLAB Computer Vision Toolbox Provides us with the toolbox supports single, 3D reconstruction and 3D point cloud processing.

Create location point cloud:

```
model_ref = zeros(y_img_choix,x_img_choix,3);
% model_ref_Color = zeros(128,170,3);
 y_img_choix_1 = y_img_choix - 1 ;
%---------- x location
   for i_test=0:y_img_choix_1 ;
         for j_test=1:x_img_choix;
                        model_ref(i_test+1,:,1 ) =  i_test;
         end
   end
%---------- y location
   for i_test=0:y_img_choix_1 ;
         for j_test=1:x_img_choix;
                        model_ref(i_test+1,j_test ,2 ) =  j_test;
```

```
            end
    end
%------------ z location  %  model_ref(:,: ,3) =  img_hsv( :, :,1) ;
parc_z = 1;
    for j_test=1:x_img_choix;
        for i_test=0:y_img_choix_1 ;
                            rad_img = img_depth_H(parc_z);
                            rad_img = round(rad_img,3);
                            Depth_rad = rad_img * 360 ;
                            if(Depth_rad <300)
                               Depth = Depth_rad ;
                            else
                               Depth = 360- Depth_rad ;
                            end
                              model_ref(i_test+1,j_test ,3) =  Depth/3;
                parc_z = parc_z + 1 ;
        end
    end
```

Listing 4.4: location point cloud.

Adding colors to the point clouds:

```
ptCloud = pointCloud(model_ref( :,:,:), 'Color', img_rgb( :,:,:));
```

Listing 4.5: colors point cloud.

4.2.2.2.4. Alignment of two consecutive frames

This stage is because we don't have on the depth sensors of camera Kinect. we fetch save dataset in Matlab in MATLAB Computer Vision Toolbox :

```
%----------------- load exemple for matlab ---------------------------
dataFile = fullfile(toolboxdir('vision'), 'visiondata', 'livingRoom.mat');
load(dataFile);
```

Listing 4.6: load dataset example a camera Kinect.

We apply an alignment algorithm ICP :

```
ptCloudRef = livingRoomData{1};
ptCloudCurrent = livingRoomData{2};

gridSize = 0.1;
fixed = pcdownsample(ptCloudRef, 'gridAverage', gridSize);
moving = pcdownsample(ptCloudCurrent, 'gridAverage', gridSize);

% Note that the downsampling step does not only speed up the registration,
% but can also improve the accuracy.

tform = pcregrigid(moving, fixed, 'Metric','pointToPlane','Extrapolate', true);
ptCloudAligned = pctransform(ptCloudCurrent,tform);

mergeSize = 0.015;
ptCloudScene = pcmerge(ptCloudRef, ptCloudAligned, mergeSize);
accumTform = tform;
% Set the axes property for faster rendering
hAxes.CameraViewAngleMode = 'auto';
hScatter = hAxes.Children;

for i = 3:length(livingRoomData)
    ptCloudCurrent = livingRoomData{i};

    % Use previous moving point cloud as reference.
    fixed = moving;
    moving = pcdownsample(ptCloudCurrent, 'gridAverage', gridSize);

    % Apply ICP registration.
    tform = pcregrigid(moving, fixed, 'Metric','pointToPlane','Extrapolate', true);

    % Transform the current point cloud to the reference coordinate system
    % defined by the first point cloud.
    accumTform = affine3d(tform.T * accumTform.T);
    ptCloudAligned = pctransform(ptCloudCurrent, accumTform);
```

```
    % Update the world scene.
    ptCloudScene = pcmerge(ptCloudScene, ptCloudAligned, mergeSize);

    % Visualize the world scene.
    hScatter.XData = ptCloudScene.Location(:,1);
    hScatter.YData = ptCloudScene.Location(:,2);
    hScatter.ZData = ptCloudScene.Location(:,3);
    hScatter.CData = ptCloudScene.Color;
    drawnow('limitrate')
end

% During the recording, the Kinect was pointing downward. To visualize the
% result more easily, let's transform the data so that the ground plane is
% parallel to the X-Z plane.
angle = -pi/10;
A = [1,0,0,0;...
     0, cos(angle), sin(angle), 0; ...
     0, -sin(angle), cos(angle), 0; ...
     0 0 0 1];
ptCloudScene = pctransform(ptCloudScene, affine3d(A));
```

Listing 4.7: ICP in matlab.

4.2.2.2.5. Save datasets of all frames

We save all dataset in an external file, to be imported with Visual Studio IDE C++ to occlusion handling:

```
fileID = fopen('C:\Users\DR.djalal\Desktop\datasetICPexemple.txt','w');

%info image
fprintf(fileID,'information_image\n');
%------ Size img refernce
```

```
[x_img_Ref,y_img_Ref,dim] = size(ptCloudRef.Location);
    fprintf(fileID,'%d\n',x_img_Ref);
    fprintf(fileID,'%d\n',y_img_Ref);
    fprintf(fileID,'information ptCloudScene.Count\n');
    fprintf(fileID,'%d\n',ptCloudScene.Count);


 %------------------ Location X Y Z : ----------------------
 % Location X :
  fprintf(fileID,'X_Location\n');
  fprintf(fileID,'%.3f\n',round(ptCloudScene.Location(:,1),3));
  fprintf(fileID,'X_Location_fin\n');
 % Location Y :
  fprintf(fileID,'Y_Location\n');
  fprintf(fileID,'%.3f\n',round(ptCloudScene.Location(:,2),3));
  fprintf(fileID,'Y_Location_fin\n');
   % Location Z :
  fprintf(fileID,'Z_Location\n');
  fprintf(fileID,'%.3f\n',round(ptCloudScene.Location(:,3),3));
  fprintf(fileID,'Z_Location_fin\n');
  %-------------------------------------------------------------
 %------------------ COLOR  R G B : ----------------------
 % Color R :
  fprintf(fileID,'R_Color\n');
  fprintf(fileID,'%d\n',ptCloudScene.Color(:,1));
  fprintf(fileID,'R_Color_fin\n');
 % Color G :
  fprintf(fileID,'G_Color\n');
  fprintf(fileID,'%d\n',ptCloudScene.Color(:,2));
  fprintf(fileID,'G_Color_fin\n');
 % Color B :
  fprintf(fileID,'B_Color\n');
  fprintf(fileID,'%d\n',ptCloudScene.Color(:,3));
  fprintf(fileID,'B_Color_fin\n');
  %-------------------------------------------------------------
```

```
fclose(fileID);
```

Listing 4.8: Save datasets of all frames.

4.2.2.2.6. adding virtual object

After importing the real scene dataset in Visual Studio IDE C++

We add an object to the scene using OpenGL:

```
initobj ("model\\Vase.obj");
obj_vase = objfile;
```

Listing 4.9: add a virtual Vase.

4.2.2.2.7. Occlusion handling

We determine the position of the camera coordinates and the viewing frame:

```
  gluLookAt(px,py,pz, px+sin(angle),py,pz+cos(angle),0.0,1.0,0.0);
```

Listing 4.10: the camera coordinates.

We project the coordinates of Z to the view frame of two objects. And with the help of a library OpenGL, we draw the result of occlusion handling:

```
//------- obj RealScene --------------------
 glPushMatrix();
                    RealScene_3DReconstruction();
glPopMatrix();
//------- obj vase --------------------
 glPushMatrix();
                    glTranslatef(-4.0,5.0,7.0);
                    glColor3f(1.0,0.0,0.0);
                    glScalef(0.2,0.2,0.2);
                    RenderOBJModel (&obj_vase);
glPopMatrix();
```

Listing 4.11: draw the result of occlusion handling.

4.2.2.3. Developing the Front-end

In the front-end part of the system, we gonna create a multi-pule user interface pages, we want to create a simple design that can every user with different gender and ages to use it, with some simple colors, and structure. The figure 4.16 and figure 4.17 shows the architecture of the front-end pages in two programs.

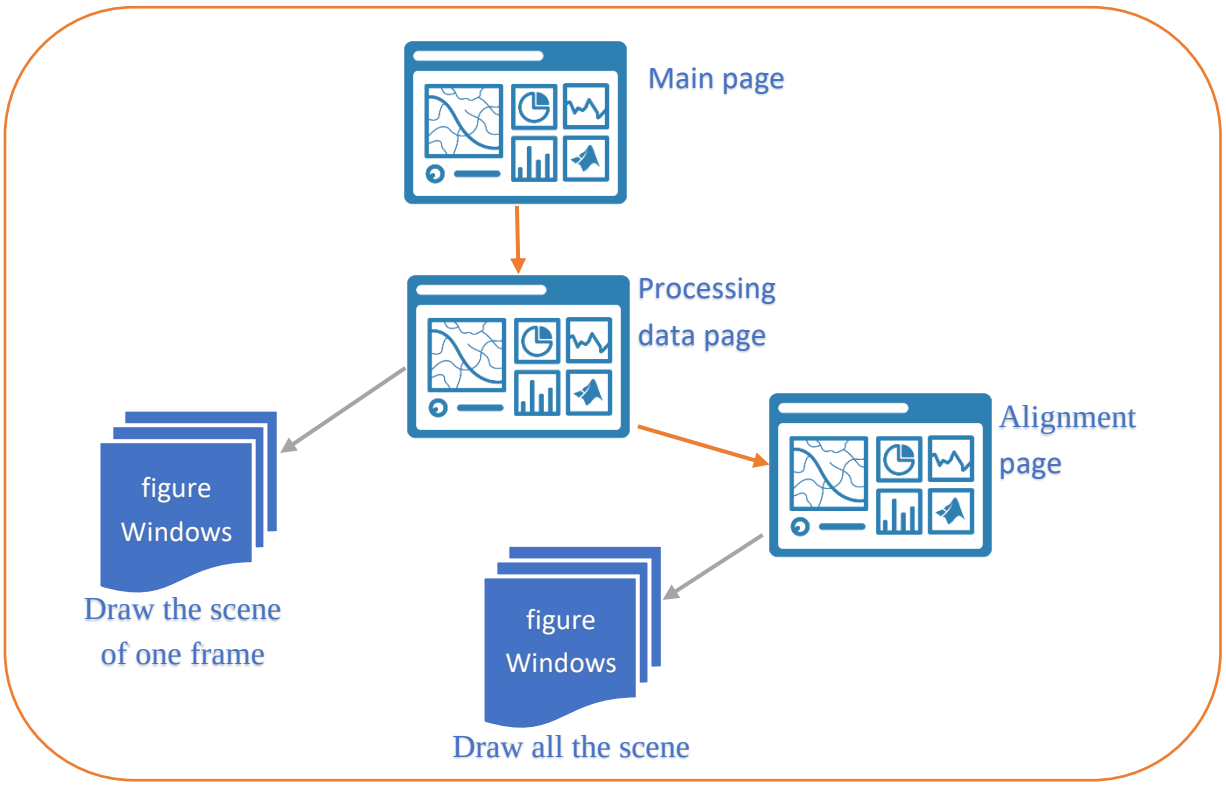


Fig 4.16: shows the architecture of the front-end pages in MATLAB.

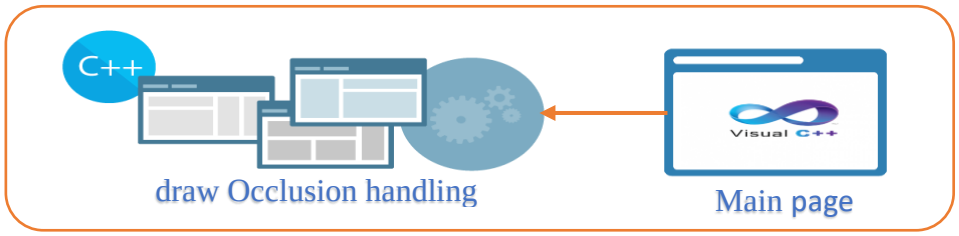


Fig 4.17: shows the architecture of the front-end pages in Visual Studio.

First, working on MATLAB for the content and the structure of the things that we want to put in out pages, With the use of the MATLAB GUI which give us a big advantage of designing and styling our page for better looking and well organizing, He also makes our pages moving and working giving some benefits on animations, button functions, pages function, Processing, display, etc..., the figures below shows the content of each created MATLAB GUI page.

Fig 4.18: MainPage.

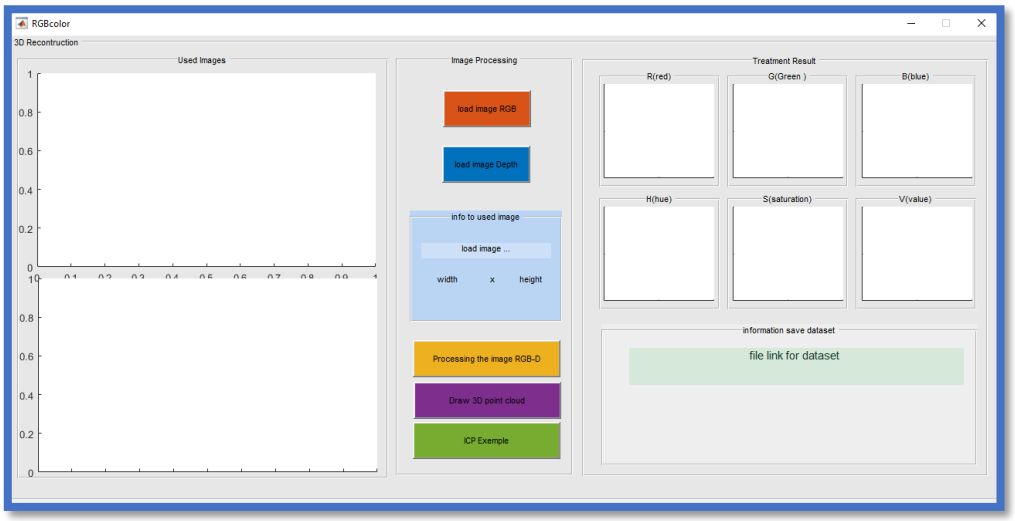


Fig 4.19: Processing data page.

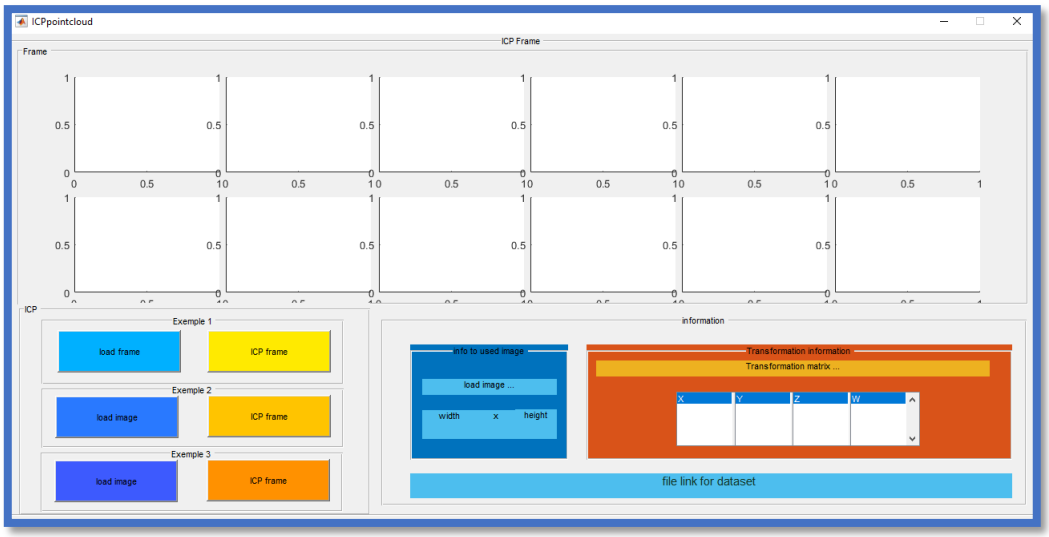


Fig 4.20: Alignment page.

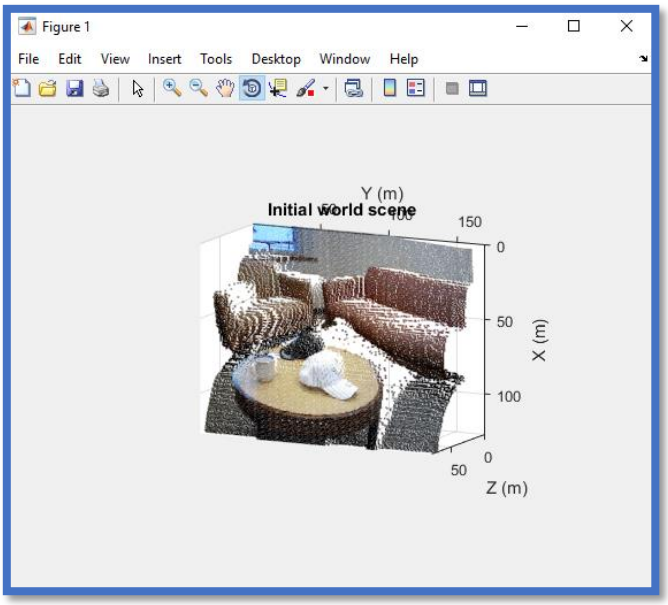


Fig 4.21: figure Windows.

second, working on Visual Studio for the final results, With the use of the OpenGL Library which give us a big advantage on draw 3D dimension, He also makes our pages moving and working giving some benefits on animations, button functions, pages function, Processing, display, etc..., the figures below shows the content of each created Visual Studio Main page.

Result pages can be selected from the list for Clicking the right button of the mouse see fig 4.22

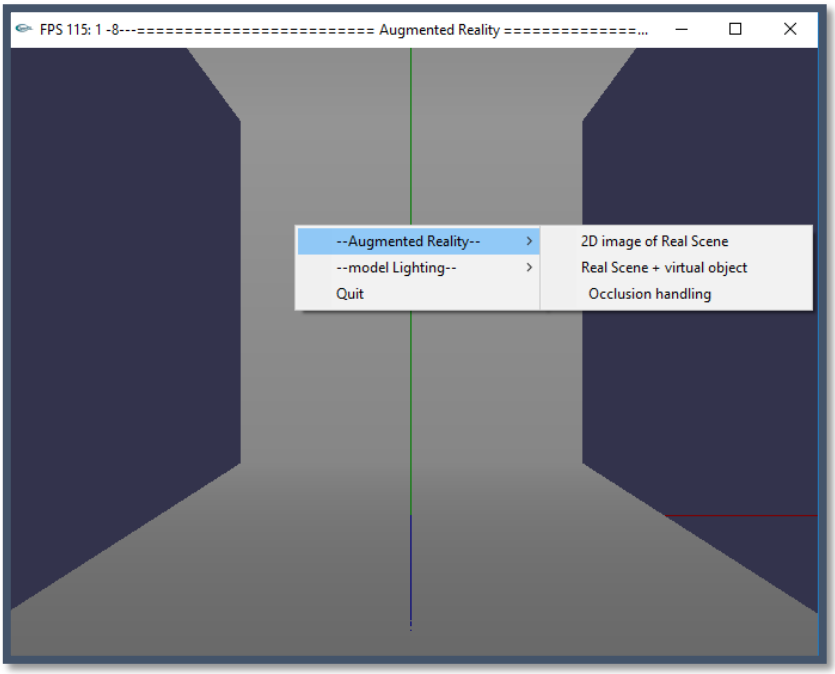


Fig 4.22: Main page.

4.3.Results

4.3.1. Photogrammetry for 3D reconstruction method

The proposed approach has been implemented using Unity and Meshroom on a laptop PC with an Intel(R) core (TM) i7-6500U CPU 2.50GHz processor and a GeForce 930M graphic cards and 8 GB RAM.

4.3.1.1. Photogrammetry

After filming the study scene, we upload the pictures to Meshroom program. After more than 4 working hours for the program, the result will appear as in the figure 4.23. Output 3D model from the photogrammetry see fig 4.24.

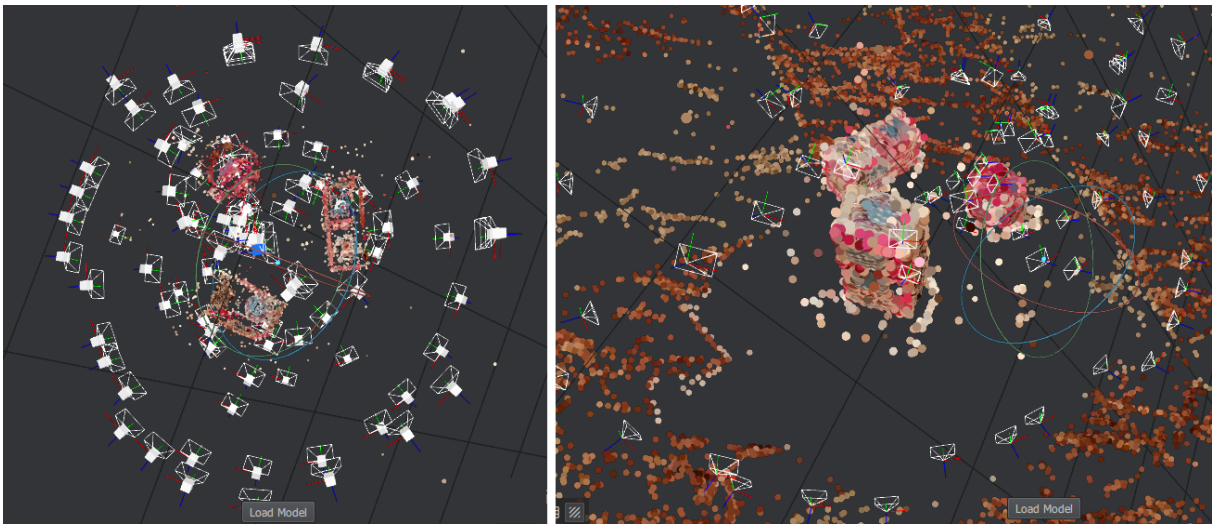


Fig 4.23: The results of photogrammetry on Meshroom.

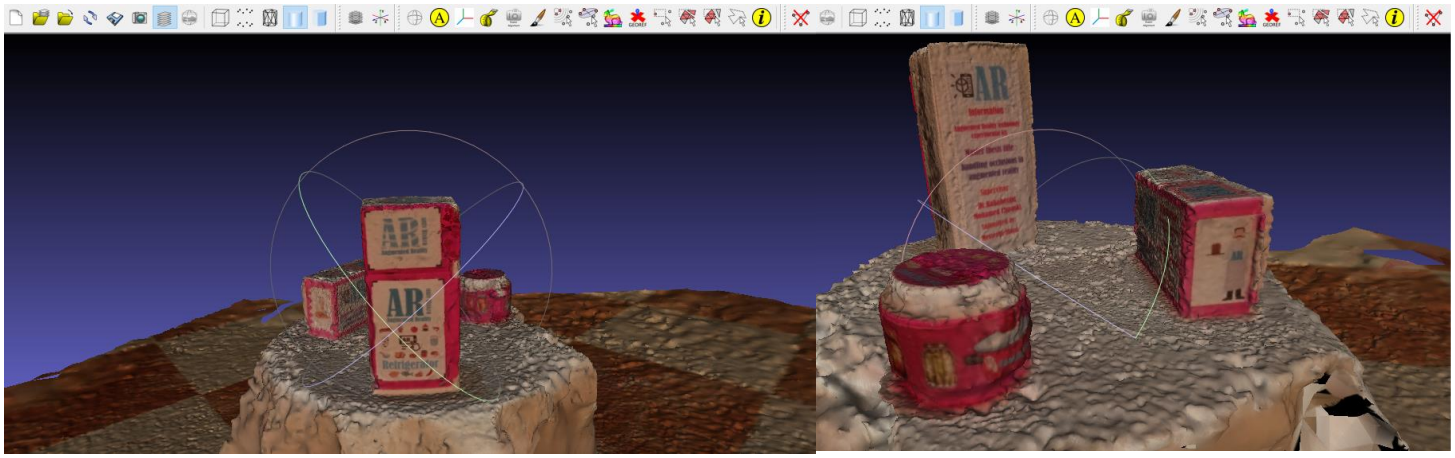


Fig 4.24: 3D Model of the study scene.

4.3.1.2. Virtual Environment

The results of the virtual environment that we created based on the study scene 3D model see fig 4.25.

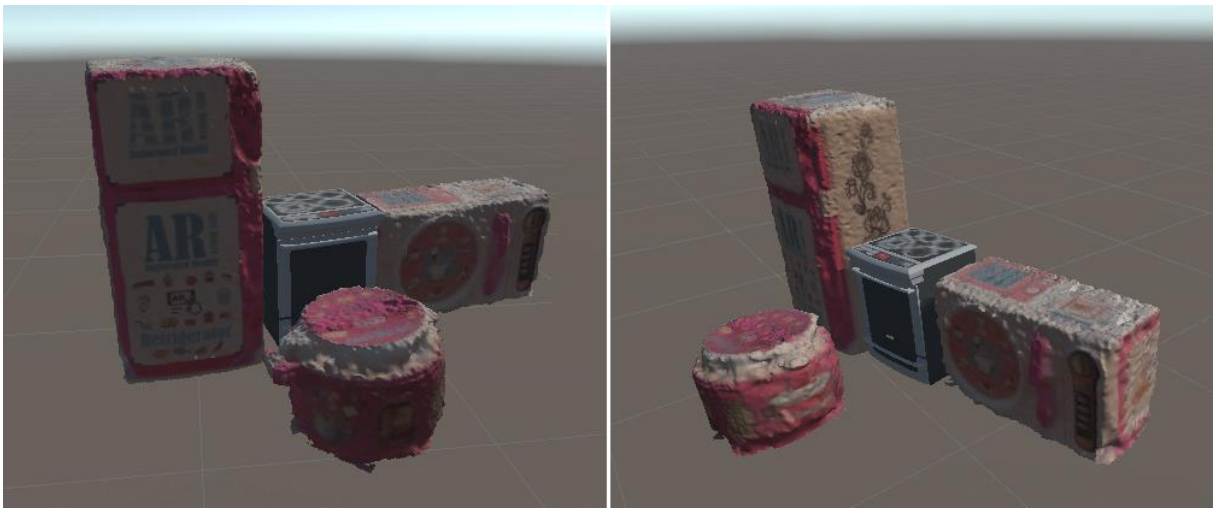


Fig 4.25: Create a virtual environment.

4.3.1.3. Real Environment

Results of the real environment in which we receive information from the camera in real time see fig 4.26.

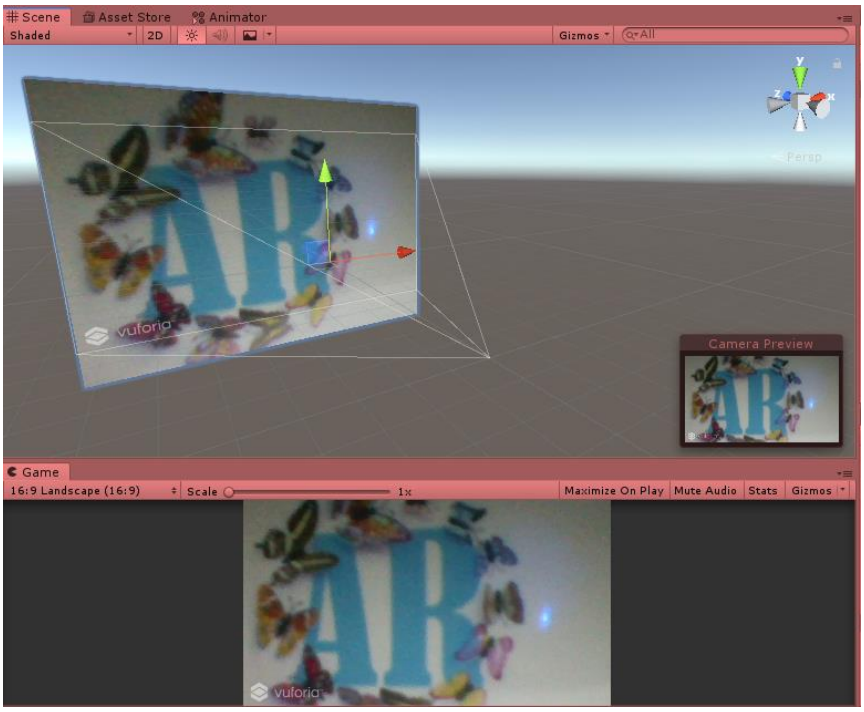


Fig 4.26: Create a real environment.

4.3.1.4. Camera alignment for AR

The results of aligning the two virtual real cameras are ARCamera see fig 4.27.

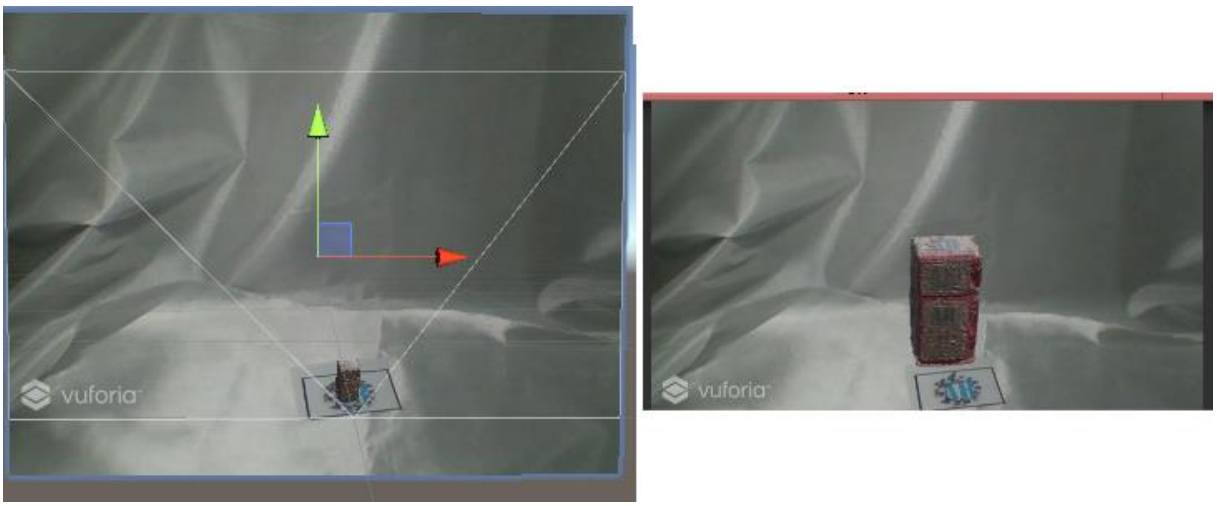


Fig 4.27: Result of ARCamera.

4.3.1.5. Occlusion handling

The image of Fig 4.28 (a) shows the result augmented reality application for tracking image. The image of Fig 4.28 (b) shows the incorrect relative relationship between the real and virtual objects without occlusion handling. The correct occlusion handling results after using our method are shown in the image of Fig4.28 (c). From the results, we can see that our method can effectively solve the occlusion problem in AR.

(a) : AR image (b) : AR Occlusion (c): AR handling

Fig 4.28: Results of the experiment to test the proposed occlusion handling method.

4.3.2. Real-time 3D reconstruction method

The proposed approach has been implemented using Visual Studio 2010 C++ and MATLAB R2016a on a laptop PC with an Intel(R) core (TM) i7-6500U CPU 2.50GHz processor and a GeForce 930M graphic cards and 8 GB RAM.

4.3.2.1. Result Processing dataset for an RGB-D camera Kinect

These images sequences are captured in real time using an RGB-D camera named Kinect for Xbox 360. All captured frames are of size (170 x 128) (pixel x pixel). The camera's intrinsic parameters of RGB camera and infrared camera are all adjust in advance using the GML (Graphics and Media Lab) calibration toolbox. Several experiments have been done to test the proposed occlusion handling method.

We process the images on the left and the result on the right in fig 4.29

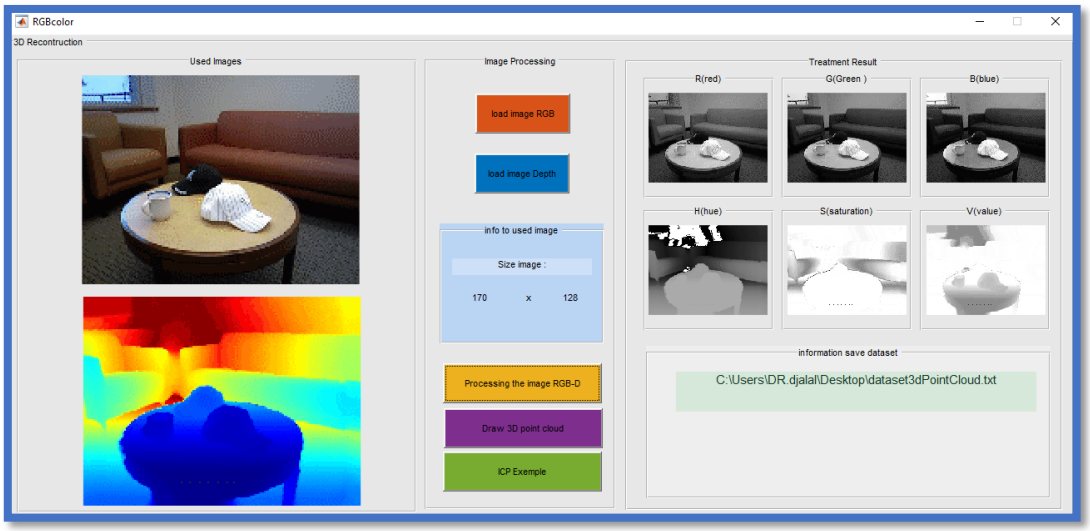


Figure 4.29. program MATLAB convert image RGBD to dataset.

4.3.2.2. Draw 3D Point Clouds

We draw the result of image processing on a figure 3d point clouds see fig 4.30.

4.3.2.3. Alignment of two consecutive frames (ICP) Results

We choose one of the examples for dataset RGB-D camera Kinect. And we apply alignment of two consecutive frames technique of ICP algorithm see fig 4.31.

4.3.2.4. Save datasets of all frames

The result is saved to a path see fig 4.32 in part of information save dataset.

4.3.2.5. Real scene drawing in Visual Studio

We import the dataset and draw the 3D scene see fig 4.33.

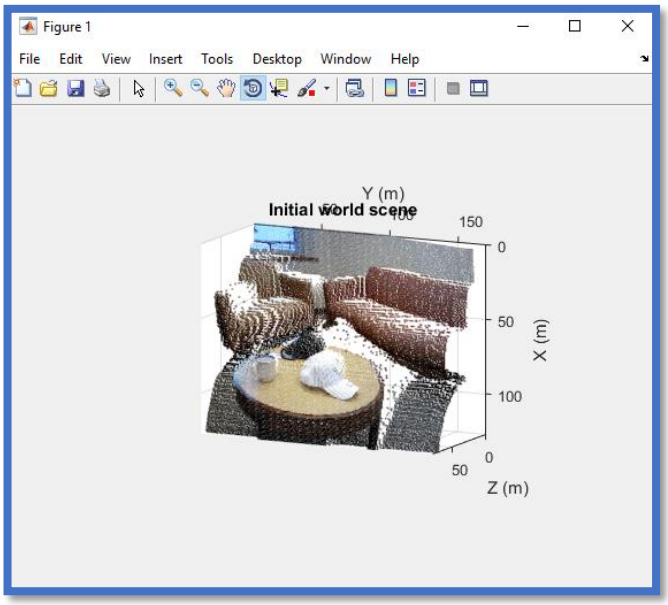


Figure 4.30. the result of figure 3d point clouds.

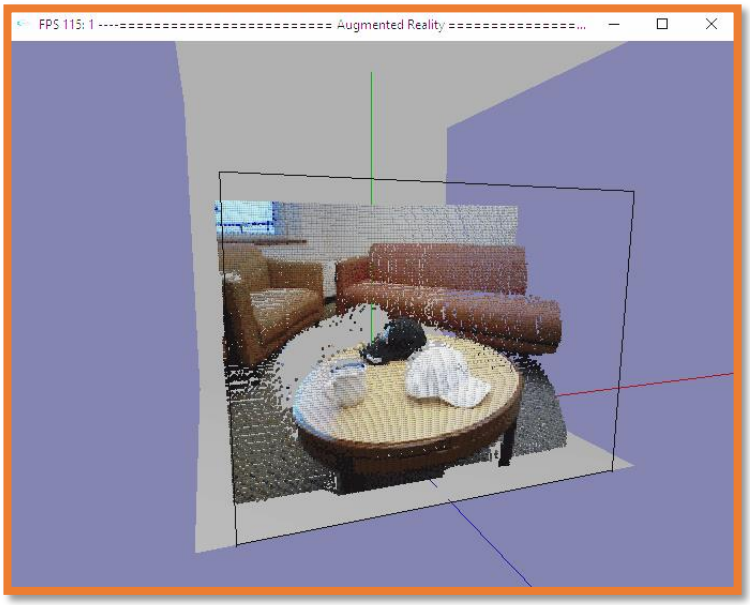


Figure 4.32. Real scene drawing in Visual Studio.

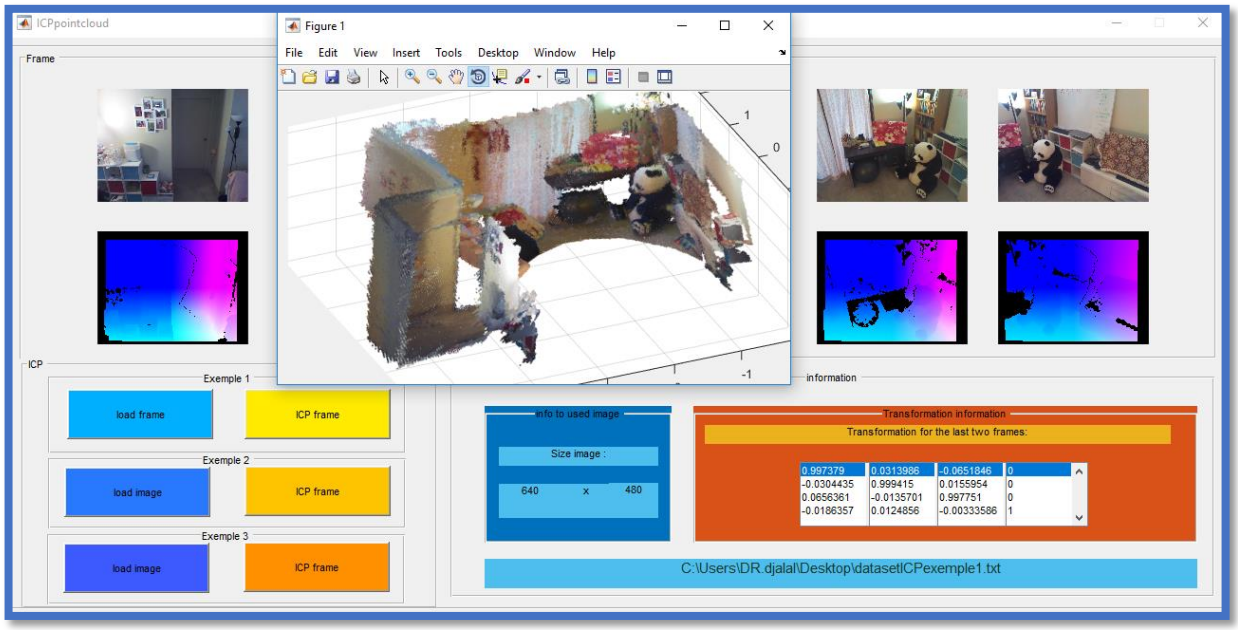


Figure 4.31. the result alignment of All frames (ICP).

4.3.2.6. Occlusion handling

In the experiment, the virtual object (vase) is a target. We overlaid the target behind the cup on a table as shown in Fig 4.33.

We obtained the 3D model of the real scene. The reconstruction result is shown in fig 4.32.

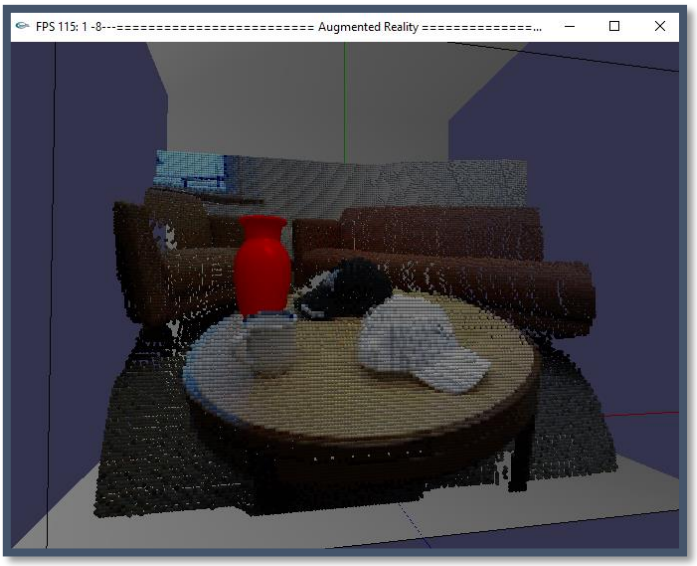


Figure 4.33. Real scene and adding virtual object.

The top images of Fig 4.34 (a)–(c) shows the incorrect relative relationship between the real and virtual objects without occlusion handling. The correct occlusion handling results after using our method are shown in the bottom images of Fig4.34 (a)–(c). From the results, we can see that our approach is effective when the camera changes over a wide range of viewing angles and volumes. This indicates that our method can effectively solve the occlusion problem in AR.

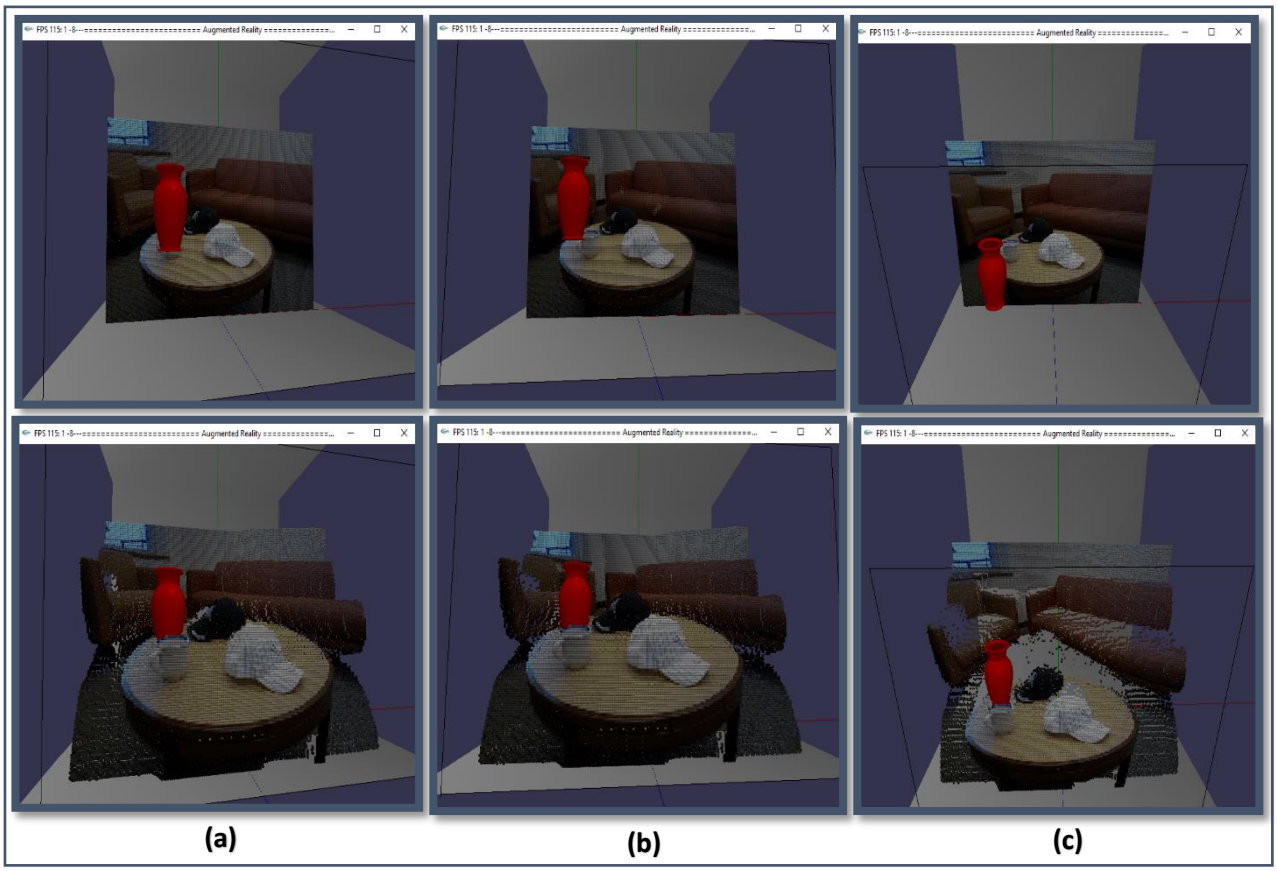


Figure 4.34. Results of the experiment to test the proposed occlusion handling method.

4.4.Discussion of results and comparison

We were able to build two systems to occlusion handling. In photogrammetry for 3D reconstruction method, we created a study scene and applied the system to it, with many tests we made we've gained the best results we could. In real-time 3D reconstruction method, we used dataset for the kinect sensor and applied the system to it, with many tests we made we've gained the best results we could.

Through the results of the two methods, we find that each of them has advantages and disadvantages as the table 4.1 shows.

Table 4.1: Comparison of the advantages and disadvantages of the two methods.

Method handling occlusions	**photogrammetry for 3D reconstruction**	**real-time 3D reconstruction**
Advantages	— It does not require expensive and expensive devices to obtain satisfactory results. — Modeling virtual objects similar to reality in shape and size. — Running augmented reality on all Android versions higher than 4.4.	— Fast in the 3D reconstruction process. — handling occlusions for any scene in real time.
Disadvantages	— Long time spent in the modeling process. — The narrow application range, since this is only suitable for static real scenes.	— Requires expensive depth sensors.

4.5.Conclusion

In this chapter, we have represented the implementation of the two methods, where we have shown the environment and the development tools we used. Then, test the effectiveness of the two methods in occlusion handling. like we also presented the graphical interfaces of two methods.

Finally, we have explained the experiments and the results obtained and compare the two methods, and we could deduce that the two methods are efficient and more accurate.

General Conclusion

The Augmented reality has become a destination for people as it provides them with an enjoyable experience and Interact with virtual objects. The Augmented reality now offer the user a real experience of the product and the service without the cost of buying it and help in providing a service without evidence. Due to the proliferation of cell phones, augmented reality is getting a lot of attention due to its importance to many companies seeking to provide a better experience for their services.

Despite seeking to provide the best service for augmented reality, the problem of occlusion stands in front of researchers.

In the first method, photogrammetry for 3D reconstruction method: a 3D model study scene is created and combined with virtual objects, and study the Occlusion and get an excellent result in occlusion handling.

In the second method, Real-time 3D reconstruction method: the study scene is captured by depth sensors in real-time, and use the data from the sensor to create a 3D model and combined with virtual objects, and study the Occlusion and get an excellent result in occlusion handling, and which encourages us to improve our research more and more.

For perspectives and future work, we offer ideas that can improve methods handling occlusions in augmented reality, such as:

— Increase the realism of virtual objects by adding shader to them.
— Add deep learning to know the sources of lighting in the scene.
— The study of lighting and its places in the scene and its effect on virtual objects.
— Merging augmented reality with virtual reality to create a mixed reality.

Bibliography

[1] Augmented reality. (2019, October 23). Retrieved December 02, 2019, from https://en.wikipedia.org/wiki/Augmented_reality.

[2] Milgram P, Kishino AF (1994) Taxonomy of mixed reality visual displays (http://vered.rose.utoronto.ca/people/paul_dir/IEICE94/ieice.html). IEICE Transactions on Information Systems E77-D (12) :1321–1329.

[3] Zhou F, Duh HBL, Billinghurst M (2008) Trends in augmented reality tracking, interaction and display:a review of ten years of ISMAR.

[4] Lee B, Chun J (2010) Interactive manipulation of augmented objects in marker-less AR using vision-based hand interaction. itng, pp.398–403, 2010 Seventh International Conference on InformationTechnology.

[5] Ababsa F, Mallem M (2008) Robust camera pose estimation combining 2D/3D points and lines tracking. Industrial Electronics, 2008. ISIE 2008. Retrieved December 05, 2019, from https://ieeexplore.ieee.org/document/4676964.

[6] Huang Y, Liu Y, Wang Y (2009) AR-View: and Augmented Reality Device for Digital Reconstruction of Yuangmingyuan", IEEE International Symposium on Mixed and Augmented Reality.

[7] Nilsson J, Odblom ACE, Fredriksson J, Zafar A, Ahmed F (2010) Performance evaluation method for mobile computer vision systems using augmented reality. IEEE Virtual Reality.

[8] Vuforia: Market-Leading Enterprise AR. (2019, November 27). Retrieved December 17, 2019, from https://www.vuforia.com/.

[9] Vuforia Developer Portal .(2019, November 27). Retrieved December 17, 2019, from https://developer.vuforia.com.

[10] Vuforia SDK Native - Android & iOS API Reference .(2019, November 27). Retrieved December 17, 2019, from https://library.vuforia.com/content/vuforia-library/en/getting-started/overview.html.

[11] Lepetit V (2008) On computer vision for augmented reality. Ubiquitous Virtual Reality, 2008. ISUVR 2008. International Symposium on, vol., no., pp.13–16, 10–13 July.

[12] Microsoft HoloLens: Mixed Reality Technology for Business. (2020). Retrieved January 04, 2020, from https://www.microsoft.com/en-us/hololens.

[13] Wagner D, Schmalstieg D handheld augmented reality displays. Graz University of Technology, Austria.

[14] Bimber O, Raskar R, Inami M (2007) Spatial Augmented Reality. SIGGRAPH 2007 Course 17 Notes.

[15] Mistry P, Maes P, Chang L (2009) WUW–Wear Ur World–A Wearable Gestural Interface",ACM, CHI 2009, Boston, April 4-9.

[16] Mills, C. (2013, March 26). Hands-On With the Iron Man Virtual Reality Setup: This is as Badass as Kinect Gets. Retrieved January 08, 2020, from https://www.gizmodo.co.uk/2013/03/hands-on-with-the-iron-man-virtual-reality-setup-this-is-as-badass-as-kinect-gets/.

[17] Schmalstieg D, Fuhrmann A, Hesina G, Zsalavari Z, Encarnacao LM, Gervautz M, Purgathofer W (2002) The studierstube augmented reality project. Massachusetts Institute of Technology 11(1):33–54.

[18] Feldman A, Tapia EM, Sadi S, Maes P, Schmandt C (2005) ReachMedia: On-the-move interaction with everyday objects. iswc, pp.52–59,Ninth IEEE International Symposium on Wearable Computers (ISWC‘05),2005.

[19] Lee J-Y, Lee S-H, Park H-M, Lee S-K, Choi J-S, Kwon J-S (2010) Design and implementation of a wearable AR annotation system using gaze interaction. Consumer Electronics (ICCE), 2010 Digest of Technical Papers International Conference on. vol., no., pp.185–186, 9–13 Jan.

[20] Li Y-bo, Kang S-p, Qiao Z-h, Zhu Q (2008) Development actuality and application of registration technology in augmented reality. Computational Intelligence and Design, 2008. ISCID‘08. International Symposium on. Vol.2, No., pp.69–74, 17–18 Oct. 2008.

[21] Papagiannakis G, Singh G, Magnenat-Thalmann N (2008) A survey of mobile and wireless technologies for augmented reality systems. Computer Animation and Virtual Worlds 19 (1):3–22.

[22] Kevin Bonsor, Nathan Chandler. (2001, February 19). How Augmented Reality Works. Retrieved January 12, 2020, from https://computer.howstuffworks.com/augmented-reality.htm.

[23] Bajarin, T. (2017, January 31). How Augmented Reality Will Change the Way You Use Computers. Retrieved January 12, 2020, from https://time.com/4654944/this-technology-could-replace-the-keyboard-and-mouse/.

[24] This AR app lets you preview furniture before buying. (2017, September 25). Retrieved January 19, 2020, from https://www.deccanchronicle.com/technology/in-other-news/250917/this-ar-app-lets-you-preview-furniture-before-buying.html.

[25] Barakonyi I, Fahmy T, Schmalstieg D (2004) Remote collaboration using Augmented Reality Video Conferencing. Proceedings of Graphics Interface. p.89–96, May 17–19, 2004, London, Ontario, Canada.

[26] Staff, H. (2017, January 27). CAE Healthcare Unveils First Mixed Reality Ultrasound Simulation Solution. Retrieved January 23, 2020, from https://hitconsultant.net/2017/01/30/cae-healthcare-ultrasound-simulation/.

[27] Mini. (2018, March 27). Official Homepage. Retrieved January 29, 2020, from https://www.mini.com/.

[28] Cool: Augmented Reality Advertisements. (2008, December 19). Retrieved January 29, 2020, from https://geekologie.com/2008/12/cool-augmented-reality-adverti.php.

[29] Augmented Reality Games. (2020). Retrieved February 02, 2020, from https://www.augmented-reality-games.com/.

[30] Marco Sacco, Stefano Mottura, Luca Greco, Giampaolo Viganņ, Institute of Industrial Technologies and Automation, National Research Council, Italy.

[31] Malaka R, Schneider K, Kretschmer U (2004) Stage-based augmented edutainment. Smart Graphics, pp.54–65.

[32] Bruna E, Brombach B, Zeidler T, Bimber O (2007) Enabling mobile phones to support large-scale museum guidance. Multimedia, IEEE 14(2):16–25.

[33] Miyashita T, Meier P, Tachikawa T, Orlic S, Eble T, Scholz V, Gapel A, Gerl O, Arnaudov S,Lieberknecht S (2008) An augmented reality museum guide. Mixed and Augmented Reality,2008. ISMAR 2008. 7th IEEE/ACM International Symposium on, vol., no., pp.103–106, 15–18 Sept.

[34] Bichlmeier C, Wimmer F, Heining SM, Navab N (2007) Contextual anatomic mimesis: hybrid in-situ visualization method for improving multi-sensory depth perception in medical augmented reality.

[35] Mangur, V. (n.d.). 25 Best Augmented Reality Games for Android and iOS. Retrieved February 13, 2020, from https://thinkmobiles.com/blog/best-augmented-reality-games/.

[36] Stutzman B, Nilsen D, Broderick T, Neubert J (2009) MARTI: Mobile Augmented Reality Tool for Industry. Computer Science and Information Engineering, 2009 WRI World Congress on, vol.5, no.,pp.425–429, March 31 2009-April 2.

[37] Elizarov, R. (2018, January 25). The Restaurants Get The Biggest Benefits in 2018 with Augmented Reality. Retrieved February 16, 2020, from https://augmentedrealityusa.com/blog-augmented-reality/the-restaurants-get-the-biggest-benefits-in-2018-with-augmented-reality/99.

[38] Fuhrmann, A.; Hesina, G.; Faure, F.; Gervautz, M. Occlusion in collaborative augmented environments. Comput. Graph.-UK1999, 23, 809-819.

[39] S. Slater and H. Childs, "Photorealistic rendering utilizing close-range photogrammetry," Phd. dissertation, University of Oregon, 2016.

[40] J. Braybon, "Traversing the unsw campus using terrestrial photogrammetry" Ph.D. dissertation, UNSW School of Surveying and Spatial Information Systems,Sydney , 2011.

[41] P. Peji´c, S. Krasi´c, H. Krsti´c, M. Dragovi´c, and Y. Akbiyik, "3d virtual modelling of existing objects by terrestrial photogrammetric methods-case study ofbarutana,"Tehni˘cki vjesnik, vol. 24, no. 1, pp. 233–239, 2017.

[42] H. J. Koelman, "Application of a photogrammetry-based system to measure and re-engineer ship hulls and ship parts: An industrial practices-based report" Computer-Aided Design, vol. 42, no. 8, pp. 731 – 743, 2010.

[43] C. Duke, "Evaluation of photogrammetry at different scales," Ph.D. dissertation,Auburn University, 2018.

[44] S. Gerbino, M. Martorelli, F. Renno, D. Speranzaet al., "Cheap photogrammetry versus expensive reverse engineering techinques in 3d model acquisition and shape reconstruction," inDS 32: Proceedings of DESIGN 2004, the 8th International Design Conference, Dubrovnik, Croatia, 2004, pp. 749–754.

[45] X. Jing, C. Zhang, Z. Sun, G. Zhao, and Y. Wang, "The technologies of close-range photogrammetry and application in manufacture," in3rd International Conference on Mechatronics, Robotics and Automation.Atlantis Press, 2015,pp. 1–7.

[46] S. F. El-Hakim, P. Boulanger, F. Blais, and J. A. Beraldin, "System for indoor 3d mapping and virtual environments" in Video metrics V, vol. 3174. International Society for Opticsand Photonics, 1997, pp. 21–36.

[47] F. Remondinoand S. El-Hakim, "Image-based 3d modelling: a review", The Photogrammetric Record, vol. 21, no. 115, pp. 269–291, 2006.

[48] G. Kortaberria, U. Mutilba, E. Gomez-Acedo, A. Tellaeche, and R. Minguez, "Accuracy evaluation of dense matching techniques for casting part dimensional verification" in Sensors, vol. 18, no. 9, September 2018, pp. 30–74.

[49] M. G. Guerra, "Analysis of a 3d optical scanner based on photogrammetry suitable for industrial applications in close and micro-range," Ph.D. dissertation, Politecnico di Bari, 08 2018.

[50] F. Grski, R. Kuczko, W.and Wichniarek, and P. Zawadzki, "Application of close- range photogrammetry in reverse engineering," in 7th International DAAAM Baltic Conference, vol. 1, 2010, pp. 30–35.

[51] A. Abdelhafiz, "Integrating digital photogrammetry and terrestrial laser scanning," Ph.D. dissertation, Assiut University, February 2009.

[52] T. Luhmann, “Close range photogrammetry for industrial applications,” ISPRS journal of photogrammetry and remote sensing, vol. 65, no. 6, pp. 558–569, 2010.

[53] H. E.-D. Fawzy, “The accuracy of mobile phone camera instead of high resolution camera in digital close range photogrammetry,” International Journal of Civil Engineering & Technology (IJCIET), vol. 6, no. 1, pp. 76–85, 2015.

[54] Max von Übel , "2020 Best Photogrammetry Software (Some are Free)". (2020). Retrieved March 05, 2020, from https://all3dp.com/1/best-photogrammetry-software/.

[55] D. P. L. FALKINGHAM, “Free photogrammetry software review:2017”. (2020). Retrieved March 05, 2020, from https://peterfalkingham.com/2017/12/17/free-photogrammetry-software-review-2017/.

[56] P. Shaffner, L. Krosley, and J. Kottenstette, “Us bureau of r eclamation digital photogrammetry r esearch r eport,” 2004.

[57] MicMac, "Gravillons tutorial". (2017, October 05). Retrieved March 9, 2020, from https://micmac.ensg.eu/index.php/Gravillons_tutorial.

[58] MicMac, "Gravillons tutorial". (2017, October 05). Retrieved March 9, 2020, from https://micmac.ensg.eu/index.php/Gravillons_tutorial#Download.

[59] A. Meshroom, “Meshroom documentation”(2020). Retrieved March 19, 2020, from https://readthedocs.org/projects/meshroom-manual/downloads/.

[60] Hable, J. (2018, August 11). Command-Line Photogrammetry with AliceVision (Tutorial/Guide). Retrieved April 04, 2020, from http://filmicworlds.com/blog/command-line-photogrammetry-with-alicevision/.

[61] Buelthoff, Heinrich H., and Alan L. Yuille. "Shape-from-X: Psychophysics and computation Archived 2011-01-07 at the Wayback Machine." Fibers' 91, Boston, MA. International Society for Optics and Photonics, 1991.

[62] Michael Saker Lecturer in Media and Communications, Leighton Evans Senior Lecturer in Media and Communications. (2019, October 16). Pokémon Go is not dead, it has 5m loyal players and it's changing people's lives. Retrieved January 04, 2020, from https://theconversation.com/pokemon-go-is-not-dead-it-has-5m-loyal-players-and-its-changing-peoples-lives-104095.

[63] S. Izadi, D. Kim, O. Hilliges, et al., KinectFusion: real-time 3D reconstruction andinteraction using a moving depth camera, in: Proc. of ACM Symposium on UserInterface Software and Technology, ACM, Santa Barbara, USA, 2011, pp. 559–568.

R.A. Newcombe, KinectFusion: real-time dense surface mapping and tracking,in: Proc. of IEEE International Symposium on Mixed and Augmented Reality,IEEE, Basel, Switzerland, 2011, pp. 127–136.

S.Y. Park, M. Subbarao, An accurate and fast point-to-plane registrationtechnique, Pattern Recognit. Lett. 24 (16) (2003) 2967–2976.

.

Listings