Automatic environment map registration around a local scene

Ulysse Larvy Delarivière

Université de Reims Champagne-Ardenne

École Doctorale Sciences du Numérique et de l'ingénieur

THÈSE

pour obtenir le grade de

Docteur de l'Université de Reims Champagne-Ardenne

Discipline : Computer science

présentée et soutenue

PAR

Ulysse LARVY DELARIVIÈRE

le 11/12/2019

Automatic environment map registration around a local scene

JURY

M. Nicolas PASSAT	Professeur, Université de Reims Champagne-Ardenne	*Président*
M. Rémi COZOT	Professeur, Université du Littoral Côte d'Opale	*Rapporteur*
M. Daniel MENEVEAUX	Professeur, Université de Poitiers	*Rapporteur*
M. Francesco BANTERLE	Research Staff CNR, Istituto di Scienza e Tecnologie dell'Informazione "Alessandro Faedo", Italy	*Examinateur*
Mme. Isabelle BLOCH	Professeure, Télécom ParisTech	*Examinatrice*
M. Yiorgos CHRYSANTHOU	Professeur, University of Cyprus, Cyprus	*Co-directeur*
Mme. Céline LOSCOS	Professeure, Université de Reims Champagne-Ardenne	*Co-directrice*

Résumé

Dans cette thèse, nous présentons une méthode pour recaler automatiquement une carte d'environnement avec une scène locale. Dans la littérature, de nombreuses méthodes ont besoin d'orienter la carte d'environnement pour obtenir une cohérence avec une scène locale. Cette orientation est dans la majorité des cas effectuée de manière manuelle par un utilisateur. Une caractéristique de notre approche est que nous n'avons pas besoin de créer un modèle 3D complet de la scène locale, ni d'interagir avec l'utilisateur.Nous proposons un pipeline pour créer une représentation virtuelle de la scène en utilisant nos données d'entrée. Cette représentation comprend la scène globale représentée par la carte d'environnement et la scène locale représentée par un objet de référence et son ombre. En utilisant la connaissance de la position de la source lumineuse principale sur la carte d'environnement, nous pouvons simuler l'éclairage et projeter une ombre sur le sol. Il est alors possible de comparer la forme de l'ombre calculée avec celle de l'ombre d'entrée pour récupérer la position correcte de la source lumineuse principale. L'orientation finale de la carte d'environnement est directement liée à la position de cette source lumineuse principale. Nous fournissons une évaluation de l'approche proposée en calculant deux métriques qui comparent notre estimation d'angle avec les directions réelles de référence. Notre estimation d'orientation montre que notre méthode récupère une orientation de carte d'environnement correcte.

Mots-clés : Carte d'environnement, éclairage, caméra 360, panorama, source de lumière virtuelle, image sphérique

Abstract

In this thesis, we present a method to automatically register an environment map with a local scene. In the literature, many methods need to orient the environmental map to be coherent with a local scene. This orientation is mostly done manually by a user. A characteristic of our approach is that we do not need to create a complete 3D model of the local scene nor interact with the user. We propose a pipeline to create a virtual representation of the scene using our input data. This representation includes the global scene represented by the environment map and the local scene represented by a reference object and its shadow. By knowing the position of the main light source on the environment map, we can simulate the lighting and project a computed shadow on the ground. It is possible to compare the computed shadow shape with the input one to recover the correct position of the main light source. The final orientation of the environment map is directly related to the position of this main light source. We provide an evaluation of the proposed approach by calculating two metrics that compare our angle estimate with actual ground truth directions. Our orientation estimation shows that our method recovers a correct environment map orientation. In this thesis, we are interested in real input data. The environment map and the local scene are extracted from photographs or videos, which already contain a lighting rendering. It is therefore important to orient the environment map in a way that is consistent with the existing lighting in the local scene. We propose an automatic method, to orient an environment map to a local scene. This method is driven by the behavior of light, drawing rays of light towards an object and attempting to match two shadows, one given as input and one calculated. We also use 3D data from the object we are considering. The originality is that we base our method on how light behaves in order to calculate and match shadows. By matching the shadows, we can estimate the correct position of the environment map.

Keywords: Environment map, relighting, illumination, camera 360, panorama, virtual light sources, spherical image

Acknowledgement - Remerciements

En préambule, je souhaite adresser sincèrement tous mes remerciements aux personnes qui m'ont apporté leur aide et qui ont ainsi contribué à l'élaboration de cette thèse.

Tout d'abord, je voudrai remercier les membres de mon jury de thèse pour leurs retours pertinents et bienveillants sur mon travail et pour avoir assisté à ma soutenance malgré les conditions particulières de celle-ci. Je remercie aussi mes encadrants, Céline et Yiorgos, qui ont su m'orienter dans mes recherches, me conseiller en faisant preuve de patience tout du long.

J'aimerai aussi remercier ma famille qui m'a apporté ses encouragements et son aide tout ce temps. Merci d'avoir fait déborder mon sac tous les dimanches soirs.

Merci à Marie pour son soutien constant et ininterrompu qui m'a permis d'aller de l'avant même dans les moments les plus difficiles.

Je remercie aussi pour leur solidarité ceux qui m'ont aidé au quotidien, collègues mais aussi amis :
David, Erwann, Exavérine, Francisco, Jimmy, Joël, Jonathan, Joris, Julien, Ludovic, Mohamed, Muhanad, Nicolas, Pierre et j'en oublie sûrement...

Merci aussi à N pour toutes ces heures passées ensemble qui m'ont permis de trouver la sortie.

Publications

Ulysse Larvy Delarivière, Yiorgos Chrysanthou, and Céline Loscos. Automatic environment map registration. In EUROGRAPHICS Short papers, 2019.

Communications

2018-November-16: Journées Françaises d'Informatique Graphique, GDR IGRV, AFIG, Poitiers. *Évolution des techniques d'acquisition et d'utilisation des cartes d'environnements.*

2019-February-14: Groupe de Travail Rendu du GDR IG-RV, Télécom ParisTech, Paris. *Orientation automatique de carte d'environnement.*

Contents

List of Figures

Chapter 1

Introduction

Résumé

De nos jours, de nombreux domaines de l'infographie utilisent des cartes d'environnement. Une carte d'environnement peut être considérée comme une représentation de l'environnement distant. Ces cartes, images à 360 degrés, permettent par exemple d'afficher un arrière-plan dans des applications de réalité virtuelle. Elles peuvent également simuler un éclairage de manière cohérente pour une scène locale créée virtuellement. Ces images servent de données d'entrée, contenant les informations de lumière distantes de la scène, et peuvent donc être utilisées pour améliorer la connaissance des paramètres de lumière. Une première étape consiste à orienter correctement la carte d'environnement par rapport à la scène locale. Par exemple, le soleil, source de lumière présente dans de nombreuses scènes d'extérieur, doit être placé au bon endroit pour pouvoir être pris en compte lors de l'éclairage de la scène locale. Dans la plupart des procédés de l'état de l'art, cette orientation est effectuée manuellement par l'utilisateur ou laissée de côté.

Intérêt des cartes d'environnement : Comme le montre cette section, l'intérêt de la communauté pour les cartes d'environnement a augmenté au cours du temps. La manière de les représenter et la diversité des domaines qui l'utilisent se sont diversifiées avec le temps. La figure 1.1, représente cet engouement en fonction du temps. Les différentes lignes de couleur représentent approximativement, pour chacun des axes, l'attrait de la communauté. Une diversité croissante de dispositifs permettant leur acquisition et de cas d'utilisation est montrée, illustrant ainsi le fort intérêt de la communauté pour ce domaine. Les zones de flou de chaque ligne expriment le fait qu'il n'est pas aisé de savoir quelles sont les utilisations possibles de chaque technologie. En effet, un domaine d'utilisation où une telle utilisation d'une carte d'environnement a pu être faite avant la date indiquée n'est pas à exclure. Ainsi, certaines œuvres peuvent utiliser, en totalité ou en partie, une technologie spécifique, présente ici même en dehors de la zone indiquée. Cette figure montre également qu'il existe un grand nombre de domaines, nombre allant croissant avec le temps, utilisant des cartes d'environnement et qu'il s'agit d'un outil qui sera encore utile dans un proche avenir.

Problématiques : Les problèmes détectés lors de nos travaux ont été les suivants.
Dans l'état de l'art, plusieurs méthodes utilisent des cartes d'environnement pour obtenir un résultat final ou un résultat intermédiaire d'une méthode. Seulement, cette étape, en fonction de l'importance de la carte d'environnement dans le processus (importante dans une méthode de rééclairage, moins dans un habillage d'environnement virtuel, par

exemple), est réalisée manuellement ou non effectuée. Dans plusieurs cas, cette étape est ignorée ou considérée comme secondaire.
Que ce soit dans une application d'éclairage ou dans une application purement décorative, pour plus de cohérence et en particulier pour la cohérence de l'éclairage, la carte d'environnement doit être correctement orientée. En effet, la scène lointaine et la scène locale ne sont pas déconnectées et la scène lointaine a un impact, notamment en termes d'éclairage sur la scène locale. Si la position des sources lumineuses ne correspond pas au profil de lumière de la scène locale, la cohérence sera perdue et l'immersion également en cas de décoration de l'environnement et des problèmes d'éclairage se produisant en cas d'application de réeclairement. Il est donc nécessaire d'adapter la carte d'environnement à la scène locale qui nous intéresse.

Objectifs de thèse : Dans cette thèse, nous nous intéressons aux données d'entrée réelles. La carte d'environnement et la scène locale sont extraites de photographies ou de vidéos contenant déjà un rendu d'éclairage. Il est donc important d'orienter la carte environnement en cohérence avec l'éclairage existant dans la scène locale. Notre premier objectif est de présenter les problèmes d'intérêt et d'orientation de carte d'environnement pour augmenter les résultats des applications en les utilisant dans leur méthode. Pour ce faire, nous proposons une méthode automatique, décrite plus en détails ci-dessous, pour orienter une carte d'environnement par rapport à une scène locale.

Notre objectif est également de comprendre le comportement lumineux de la scène. De nombreuses applications sont possibles avec une amélioration des résultats ou de la cohérence de ceux-ci, lorsque le profil de lumière présent dans l'environnement est adapté à la scène traitée. Comprendre la lumière d'une scène permet de mieux appréhender l'ombre d'un l'objet comme sa position ou les réflexions de lumière.

Contributions : Cette thèse présente les contributions suivantes.

Une étude sur les cartes d'environnement : Nous développons dans le chapitre 2 une étude sur les cartes d'environnement qui présente les différents aspects de cette représentation. Premièrement, nous abordons différentes techniques d'acquisition utilisées pour capturer ou obtenir une carte d'environnement. Nous présentons des méthodes à partir de sources réelles, comme la création de mosaïques à partir de photographies, l'utilisation de caméra dédiée ou d'une sphère chromée. D'autres méthodes permettent de les obtenir à partir d'informations artificielles comme la capture d'environnement virtuel ou le calcul

d'un modèle analytique.
Ensuite, l'importance des cartes d'environnement dans le domaine du rendu est décrite. Un panorama de leurs utilisation est également dressé pour démontrer l'attrait de cet outil pour la communauté. Différents domaines d'application sont mentionnés comme l'édition vidéo, la réalité augmentée ou l'apprentissage profond (deep learning). De plus, d'autres applications, en dehors du domaine purement informatique, peuvent étre mentionées, comme la visite virtuelle de lieux parfois nécessitant la création d'images 360° comptant des milliards de pixels. Enfin, nous présentons différentes méthodes nécessitant une orientation robuste des images 360° comme la navigation de véhicules robotisés dans un environnement inconnu.

Un algorithme de recalage automatique de cartes d'environnement : Nous développons dans les chapitres 3, 5 et 4 une nouvelle méthode pour orienter automatiquement une carte d'environnement autour d'une scène locale en utilisant l'ombre d'un objet de référence et l'information lumineuse présente dans la scène. Ce travail a été publié dans la session *(Eurographics short paper* en 2019). Nous exposons d'abord dans le chapitre 3 l'idée générale de la méthode et les différentes données d'entrée : la scène locale, la scène globale et les informations supplémentaires que nous avons extraites des données d'entrée. Le chapitre 5 montre les différentes tentatives que nous avons faites. Il est divisé en deux catégories : celles qui utilisent uniquement des informations 2D dans la scène locale pour récupérer l'orientation de la carte d'environnement, et celles qui visent à récupérer ou utiliser une partie de la 3D de la scène locale en utilisant des imposteurs ou l'information de profondeur. Nous exposons de plus les limites de ces méthodes et expliquons à quel point il est important d'utiliser la connaissance 3D de la scène locale afin d'obtenir des résultats cohérents.
Nous présentons, dans le chapitre 4, une méthode utilisant la géométrie d'un objet de référence présent dans la scène locale pour récupérer une orientation correcte de la carte d'environnement autour de la scène locale. On extrait alors dans un premier temps l'orientation de l'ombre de l'objet de référence, puis dans un second temps, on compare cette ombre de référence à une ombre calculée, en suivant le trajet des rayons lumineux, par calcul virtuel.
Une étape de minimisation est alors effectuée pour récupérer la meilleure orientation de la carte d'environnement. Nous exposons enfin les résultats obtenus, et les limites que nous avons identifiées. Nous proposons des améliorations possibles à effectuer dans des travaux futurs.

Méthode théorique utilisant le Machine learning : Nous présentons également dans cette thèse une approche préliminaire d'une méthode d'apprentissage pour trouver l'orientation d'une carte d'environnement. Nous exposons le travail effectué jusqu'à présent : la recherche effectuée sur la forme du réseau neuronal, la création d'un jeu de données virtuel, l'étude des méthodes existantes et plus précisément la comparaison possible avec une méthode déjà parue. Nous exposons ensuite la limitation de cette approche ainsi que la difficulté de la collection d'un ensemble de données réelles suffisantes de paires d'images (scène locale / scène globale) dans différentes expositions, lieux, moments de la journée, *etc.* Nous essayons de surmonter cette limitation par la création d'un ensemble de données virtuelles. Nous exposons alors les avantages et les inconvénients d'une telle approche.

Présentation du document : La thèse est organisée comme suit :

- Le chapitre 2 présente l'état de l'art des cartes de l'environnement. Les techniques d'acquisition, leurs utilisations en rendu et dans d'autres domaines sont présentées. Enfin, des méthodes utilisant l'orientation des cartes d'environnement est exposée.
- Dans le chapitre 3, un préambule de notre méthode de recalage automatique de carte d'environnement autour d'une scène locale est exposée. Les données d'entrée et une vue d'ensemble de notre méthode sont exposées.
- Le chapitre 4 présente notre solution, en utilisant l'information 3D d'un objet de référence présent dans la scène locale. Nous développons les différentes manières par lesquelles nous obtenons la géométrie 3D d'une scène et présentons en détail l'étape d'orientation de notre méthode. Enfin, nous exposons les résultats et présentons une conclusion et les limites d'une telle méthode.
- Le chapitre 5 présente différentes pistes testées afin de réduire le nombre d'hypothèses ou de données d'entrées de notre méthode, en particulier la connaissance de la géométrie de la scène locale. Différentes façons de définir les rayons lumineux, un pipeline 2D complet, ou des imposteurs sont utilisés. Nous présentons les résultats et discutons des limites de ces méthodes. Nous expliquons alors pourquoi une connaissance de la géométrie de la scène locale est importante.
- Le chapitre 6 résume les contributions de cette thèse et conclut. Quelques travaux futurs et les améliorations possibles sont répertoriés. Nous développons également quelques pistes que nous suivons sur une méthode de Machine learning pour orienter une carte d'environnement avec une scène locale.

Addendum : Ci-dessous, nous listons les synonymes et acronymes utilisés pour décrire le concept de *"carte d'environnement"*. Dans la littérature, les différentes communautés qui

utilisent cet outil peuvent nommer les *"cartes d'environnement"* en utilisant différentes désignations. La différence de dénomination peut venir de la technique d'acquisition (comme pour *chrome sphere* ou *mirror ball*), des techniques de représentation (comme pour *equirectangular image*), du domaine (comme pour *cube map* largement utilisé en rendu), des utilisations (comme pour *panorama* pour utilisation visuelle), *etc.* D'autres termes désignent la fonction ou les propriétés de cet outil (comme *image sphérique*, *image à 360°* ou *image grand angle*).

Certains termes sont génériques, d'autres spécifiques à un domaine, certains encore peuvent se référer à des concepts liés (comme le terme de *lumisphere* par exemple qui désigne également un matériau cellulaire fluorescent [HRP80]).

Enfin, on peut supposer que cette liste n'est pas exhaustive. En effet, en raison de la diversité des domaines utilisant les cartes d'environnement et de la diversité de leurs représentations (voir figure 1.1), le nombre de termes utilisés augmente, et nous ne pouvons pas être certains de référencer tous les termes dans cette liste.

environment map	EM	EnvMap
360 photos	360° image	180° image
panorama	spherical image	panoramic image
wide angle image	large angle image	lumisphere
miror ball	chrome sphere	cube map
omnidirectional image	omnistereo image	omnistereoscopic panorama
equirectangular image	reflectance field	reflection cards
sky map		

TABLE 1.1 – Liste de termes identifiés comme faisant référence ou étant associés aux cartes d'environement.

1.1 Context

Nowadays, several domains in computer graphics use environment maps. An environment map can be viewed as a representation of the far environment. Theses maps allow, for example, to display background in virtual reality applications. They can also simulate coherent lighting for a local virtual scene. These maps act as input data containing the distant light information of the far scene and can thus be used to improve the knowledge of the light parameters. In order to use an environment map , a first step is to correctly orient it in relation to the local scene. For example, the sun or the light source present in the distant scene, must be placed in the right place for proper consideration during the illumination of the local scene. In the most methods of the state of the art this orientation is done manually by the user or left aside.

The way to represent environment map and the diversity of domains which use it have increased with time. Figure 1.1 represents theses infatuations over time in the community. The different coloured lines represent roughly, for each axis of the state of the art, the community lure over time. A growing diversity of devices and use cases are illustrated, showcasing a strong interest of the community for this field. Blur areas correspond to the fact that this figure approximates the exact usage time of technology. In fact, some works can resort to a specific technology even outside this defined lure area. This figure also shows that there is a very diverse and large number of works using environment maps, and that is a tool which will still be useful in the near future.

1.2 Problematic

In the state of the art several methods use environment maps to obtain a result or intermediate result. This step is usualy done manually or not done at all, depending on the importance of the environment map in the process (crucial in a relighting method which aim to change the light profile of a scene. Not important in a virtual environment decoration, for example). In many cases, this step is ignored and considered secondary.
However, whether in a relighting application or in a purely decorative application, for more coherence, especially lighting coherence, the environment map must be well oriented. Indeed, the far scene and the local scene are not disconnected, and the far scene has an impact, especially in term of lighting on the local scene. If the positions of the light sources are not in adequacy with the local scene light profile, the coherence, thus, immersion may be lost and lighting problems will occur in case of relighting application.

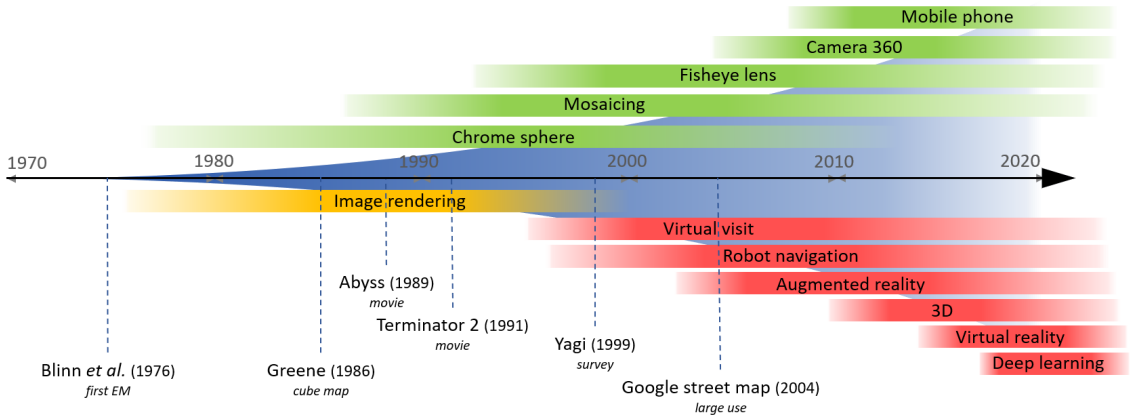


Figure 1.1 – Color lines represent roughly, for each section of the paper, the community lure over the time. Acquisition sections are shown in green, uses in yellow (for rendering) and red. Some dates are also exhibited to shown key works or relevant use of environment map.

1.3 Thesis objectives

In this thesis, we are interested in real input data. The environment map and the local scene are extracted from photographs or videos, thus already containing lighting effects. It is therefore essential to orient the environment map in coherence with the existing lighting of the local scene. Our goal is to solve environment map orientation (*i.e.*, registration) issues to guarantee that the environment lighting positioning accurately coincides with the resulting lighting effect that can be observed in the local scene. Our objective is to propose an automatic approach that requires minimal user input and scene knowledge.

1.4 Contributions

This thesis presents the following contributions:

A study about environment map: We present in chapter 2 a study about environment maps which present different aspects of this tool. First, we present different acquisition techniques used to capture or obtain an environment map. Some methods from real sources, as the stitching of photography or the use of a 360° camera or a chrome sphere. Other methods from artificial information as the capture of a virtual environment or the computation of an analytic model.
Then, the importance of environment maps in the rendering domain is described. A panorama of environment maps usage is also made to show the community lure for this

tool. Different domains of applications in computer science are mentioned as the video edition, augmented reality, or deep learning. However, also different application outside the computer science sphere or virtual visit, potentially needing the creation of 360 image counting billions of pixels. Finally, we present different methods for which robust 360° image orientation is important, as robot navigation in unknown environments.

Automatic environment map registration algorithm: We present in chapters 3, 5, and 4 a new method to automatically orient an environment map around a local scene using the shadow of a reference object and the light information present in the scene. This work was published in the *Eurographics short paper* in 2019. We first expose in chapter 3 the general idea of the method and the different input data: the local and global scene and additional information we extracted from the input data.
We present, in chapter 4, a method using the geometry of a reference object present in the local scene to recover a correct orientation of the environment map around the local scene. We first recover the orientation of the reference object's shadow and then compare it to a shadow generated, following the light path, by virtual computation using the environment map.
A minimisation step is done to recover the best orientation of the environment map. We finally expose our results and talk about the limitations and possible improvement. The chapter 5 shows our attempts to reduce hypothesis or input data of our method. It is divided into two categories: ones which use only 2D information of the local scene to recover the orientation of the environment map, and others which aim to recover or use partial 3D of the local scene using impostor or depth information. We expose the limitations of these class of methods and explain how important it is to use 3D knowledge of the local scene.

Machine learning theoretical method: We present introductory work on machine learning method to find the orientation of an environment map. We expose the work done for this thesis: the research made about the neural network shape, a virtual dataset creation, a study of the existing methods and more precisely the comparison with a state of the art method.
We then expose the limitation of this approach and the difficulties of gathering dataset of pair of images (local scene / global scene) from real scene in different expositions, places, the moments of the day, *etc.* We try to overcome this limitation by the creation of a virtual dataset. We expose the advantages and disadvantages of such an approach.

1.5 Document overview

The thesis is organized as follows:

- Chapter 2 presents a state of the art on environment maps. Acquisition techniques, image mapping, and usage are presented. Finally, environment maps positioning technique are exposed.
- In chapter 3 a preamble of our automatic environment map registration method around a local scene is exposed. Input data and an overview of our method are exposed.
- Chapter 4 presents our solution using the 3D geometry of a reference object in the local scene. We develop how the 3D geometry of the scene is recovered and show in detail the registration step of our method. Finally, we show results and present a discussion on limitations of such a method.
- Chapter 5 presents approaches we test to reduced hypotheses or input data. Differents ways to define light rays are used to develop different test methods using full 2D pipeline or impostors. We present results and discuss about limitations of those methods. Moreover, we explain why a knowledge of the geometry of the local scene is important.
- Chapter 6 summarizes the contribution of this thesis and concludes. Some future work and possible improvements are presented. We also develop some plausible directions on a machine learning method to register an environment map with a local scene.

1.6 Addendum

Bellow, we list synonyms and acronyms used to describe the *"environment map"* concept. In the literature, the different communities can name the *"environment map"* using different designations. The naming difference can come from the acquisition technique (as for *chrome sphere* or *mirror ball*), representation techniques (as for *equirectangular image*), the domain (as for *cube map* largely used in Rendering), the uses (as for *panorama* for visual uses), *etc.* Other term design the function or properties of this tool (as *spherical image*, *360° image* or *wide angle image*).

Some choices of the terms are generic and other specific to a domain. Some terms can refer to linked concepts (as the term of *lumisphere* for example which also design a fluorescent cellular material [HRP80]). Finally, we can suppose that this list is not

exhaustive. Indeed, due to the diversities of domains using environment maps and their diversity of representations (see figure 1.1), the numbers of used terms increase, and we cannot be sure to reference every term in this list.

Environment map synonyms and acronyms:

environment map	EM	EnvMap
360 photos	360° image	180° image
panorama	spherical image	panoramic image
wide angle image	large angle image	lumisphere
miror ball	chrome sphere	cube map
omnidirectional image	omnistereo image	omnistereoscopic panorama
equirectangular image	reflectance field	reflection cards
sky map		

Table 1.2 – Possible acronyms, synonyms, or associated terms of environment maps.

Glossary:

We present here a glossary of terms used in this manuscript:

- Convex hull: The smallest subset of points which include a set of points. Deduced from the projected points, we compute a convex hull to reduce the number of points for a computed shadow and obtain a comparable shadow shape to the input one.
- Depth: Depth image given as input. In this work, the depth image allows us to recover a rough 3D set of points of the reference object in chapter 4.
- EDT: Euclidean Distance Transform. A distance map, representation of the distance to a shape. We use it to compute a metric to compare two different shadow shapes.
- Environment map: 360° images. See *global scene* on this glossary. Some synonyms are presented in part 1.6 on page 31.
- Field-of-view: Represents the camera lens angle. In this document, a large field-of-view image is an image which represent a scene with an angle larger than 180° or an environment map image, and a low field-of-view image represent a local scene image.
- Geometry: 3D information of the reference object. A set of points given as input, representing the reference object and directly used to recover a computed shadow shape. Used in chapter 4.

- Global scene: Far scene surrounding the local scene, which is comparable to a background. It usually contains more information (on elements in the global scene, light information, geometry *etc.*) than low field-of-view images thanks to its larger field-of-view.

- HDR: High dynamic range. A representation of light intensities larger than low dynamic range images. Specifically, it extends the range to allow non saturated colors both in underexposed (dark) and overexposed (bright) areas. Because such an image represents a large range of intensities, it can serve as an environment lighting representation.

- Impostor: 2D representation of a 3D object. An image placed in a 3D space to represent an object. In this work, we use them to represent the reference object and its shadow in chapter 5.

- Local scene: Scene (objects, geometry *etc.*) close to the user. Usually the area we are looking for.

- Match: In chapter 5, we define a match as a correspondence between a point of the reference object, and a point of its shadow. To find correct matches, we have defined rules exposed in section 5.2.1.

- Method: We present a different method to register the environment map around a local scene:

  - A 2D method which uses only 2D input images to define rays and recover the light source direction.
  - A 2D/3D method which uses impostors to compute 3D rays and recover the light source position on the environment map surface.
  - A full 3D method which uses 3D information (depth or geometry of a reference object) to define a shadow shape and compare it to an input shadow.

  A summary table 6.1 on page 151 shows in details the differences between these methods.

- Radius: Radius of the environment map. Even if this value allows us to obtain the correct environment map orientation, it has no physical meaning because we make the assumption that the environment is placed at infinity.

- Ray: In this work, we define a light ray as a 2D or 3D line that connect a light source, an object, and the associated shadow shape. We define three types of rays in section 3.3.

- Reference object: Main central object of the local scene. The object of the local scene we use to compute and compare the shadows.

- Registration: Corresponding geometrical correct positioning between the content of an environment map and the local information of a same scene.

- Shadow: Occluded part from the light source, resulting in a dark region visible on an image. In this work, we compare a shadow shape given as input with a computed shadow shape to recover the correct environment map orientation.

- Theta: θ. The angle we are looking for to register the environment map. It corresponds to the angle that the environment map has around the z-axis. For a better understanding, see figure 3.1 on page 77.

Chapter 2

State of the art - Previous work

Résumé

Une carte d'environnement (*environment map* ou EM) est une image qui donne des informations sur la lumière entrante d'une scène. Traditionnellement, une carte d'environnement est une image sphérique à 180° ou 360°. D'autres types de représentations possibles sont une image cylindrique ou une *cube map*, cube composé de 6 faces, qui sont aussi des représentations entourant la scène. Une caméra montée avec un objectif fish-eye est l'appareil le plus couramment utilisé pour obtenir de telles images. Une autre façon de capturer une carte d'environnement est d'utiliser une sphère chromée comme dans [JNVL10] ou [Deb08]. Les appareils plus récents permettent de capturer directement des images sphériques 360°, par exemple, une caméra Ricoh Theta utilisée dans [BTH15], un dispositif regroupant quatre caméra GoProest exposé dans [LN15] ou une caméra Ladybug dans [SY10].

La diversité des représentations de ces images s'explique par la grande diversité des données d'entrée et des usages faits de cet outil. Si l'objectif est de fournir un environnement agréable pour une scène 3D (une vue d'architecte, par exemple) et non d'avoir une représentation précise de la lumière entrante, une représentation cubique est suffisante. Inversement, si le but est d'inclure des objets dans la scène pour une application de réalité augmentée, virtuelle ou mixte et de fournir une bonne immersion à l'utilisateur, une carte d'environnement sphérique complète est alors préférable.

Dans la figure 2.1, différentes représentations de cartes environnementales sont présentées. De haut en bas, de gauche à droite : une image sphérique équirectangulaire (2.1, a) capturée à l'aide d'une caméra 360 dans [BCD+13], une représentation cubique (2.1, b) et sa représentation applatie associée issue de [LK10], un panorama cylindrique (2.1, c) issu de [PSHZ+15]. Vient ensuite une carte d'environnement zenithale (2.1, d) issue de [Kán15], une image capturée à l'aide d'une sphère chromée (2.1, e) issue de [JNVL10]. La dernière ligne présente une carte d'environnement sphérique virtuelle (2.1, f) issue de [CWW11] et une image parabolique, sphérique, et une représentation sur une sphère (2.1, g) de la même scène issue de [MD06].

Les images panoramiques ou grand angle ainsi obtenues présentent un avantage par rapport aux images "classiques" (ou à faible champ de vision) dans la façon dont l'utilisateur appréhende les scènes mais aussi pour le traitement informatique. En effet, l'information est complète tant pour les éléments présents dans la scène que pour la compréhension de l'illumination. Cette caractéristique permet une meilleure compréhension de l'environne-

ment et donc de meilleures possibilités d'utilisation. Par exemple, l'insertion d'un objet dans une scène d'une manière cohérente est possible, prenant en compte la bonne direction des ombres par rapport aux objets présents dans la scène et les positions de la source lumineuse, en tenant compte de la couleur de la lumière, de la réflexion, de l'occlusion de l'objet ou des ombres de l'objet près de l'objet inséré. Sans cette connaissance, l'objet dénote par rapport au reste de la scène et le résultat est de moindre qualité.

Dans ce contexte, la scène peut être divisée en deux parties : la scène locale, qui a l'angle de vue d'une caméra et contient la région d'intérêt, et un environnement (ou scène globale) et la carte d'environnement, représentant l'environnement. Dans les différents travaux présentés dans ce manuscrit, différents termes peuvent être utilisés pour définir les images 360° : omnidirectionnelle, sphérique ou panoramique, panorama, grand angle ou grand angle. Dans ce rapport de thèse, nous préférerons le terme générique de carte d'environnement (ou *environment map* en Anglais) pour les définir.
Étant donné qu'une carte d'environnement est une image 360°, elle contient donc toutes les informations de la scène qui nous intéresse. Ces informations peuvent être diverses : informations sur le paysage, la lumière ou la géométrie de la scène. Les informations utiles dépendent de l'utilisation faite de la carte d'environnement. Un aperçu détaillé de l'utilisation des cartes de l'environnement est présenté dans la section 2.5. Dans les différentes sous-sections de cet état de l'art, l'information utile n'est pas forcément la même.
De plus, dans plusieurs utilisations d'images panoramiques, l'imagerie HDR peut être utilisée pour obtenir de meilleurs résultats. Une image HDR [RHD+10] est une image qui présente une large gamme d'intensités et peut donc être une représentation correcte pour l'éclairage. Une telle image présente des informations dans les zones sous-exposées et surexposées.

Tout d'abord, dans la section 2.1.2, nous présentons ce qu'est un panorama et de quelle manière un panorama est lié au concept de carte d'environnement. Dans la section suivante, 2.1.3, nous présentons différentes méthodes d'acquisition de cartes d'environnement. Dans la section 2.5, nous discutons des différentes utilisations des EMs. Enfin, dans la section 2.6, nous présentons des méthodes pour orienter les cartes d'environnement présentes dans l'état de l'art.

2.1 Environment map - Representation

2.1.1 Introduction

An environment map (or EM) is an image which gives information on the incoming light of a scene. Traditionally, an environment map is a 180° or 360° spherical image. Other types of possible representation are a cylindrical image or a cube-map which are also representations surrounding the scene. A camera mounted with a fish-eye lens is the most common device used to obtain such images. Another way to capture an environment map is to use a chrome-like sphere [JNVL10] [Deb08]. Other devices allow capturing directly 360° spherical images. For example, a Ricoh Theta [1] is used in [BTH15], and a device gathering four GoPro [2] cameras is exposed in [LN15] or a Ladybug [3] camera in [SY10].

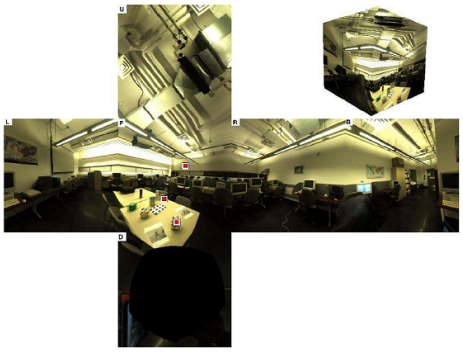
The diversity of representations of such images is explained by the great diversity of input data and their different usages. If the goal is to provide a pleasant environment for a 3D scene (an architect view, for example) but not to have a precise representation of the incoming light, a cube-map representation is sufficient. Conversely, if the goal is to include objects in the scene for an augmented, virtual, or mixed reality application and to provide better possible immersion to the user, a complete spherical environment map is preferable.

In figure 2.1, different representations of environment maps are exposed. Top to bottom, left to right: an equirectangular spherical environment map capture (2.1, a) with a 360 camera from [BCD+13], a cube map (2.1, b) and a cubic flattened representation from [LK10], a cylindrical panorama (2.1, c) from [PSHZ+15]. Then, a zenithal environment map (2.1, d) from [Kán15], a light probe image (2.1, e) from [JNVL10]. The last line present a virtual spherical environment map (2.1, f) from [CWW11] and a parabolic, spherical image, and a representation on a sphere (2.1, g) of the same scene from [MD06].

The panoramic or wide-angle images thus obtained provide an advantage over the "classical" (or low field-of-view) images in the way the user apprehends the scenes but also for the computer processing. Indeed, the information is comprehensive both for the elements present in the scene and for the understanding of illumination. This characteristic gives a better understanding of the environment and therefore, better possibilities of use. For example, the insertion of an object in a scene in a coherent way is possible, including the correct direction of shadows relatively to the objects present in the scene and light source

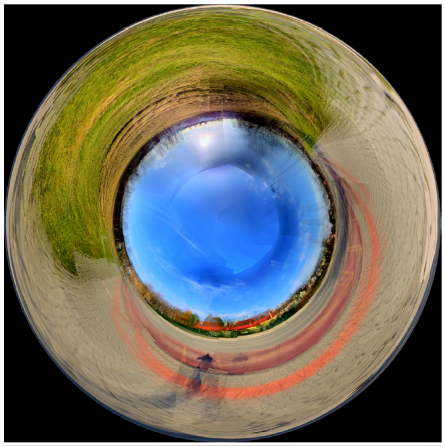
[1] Ricoh Theta website
[2] Go pro website
[3] Ladybug website

(a) Equirectangular image. [BCD+13]

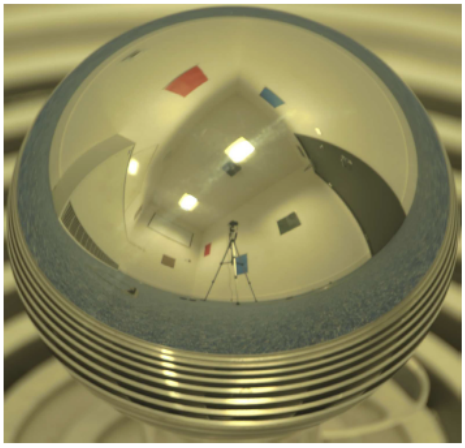


(b) Cubic representation. [LK10]

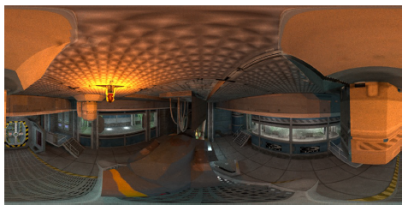
(c) Cylindrical panorama. [PSHZ+15]



(d) Zenithal representation. [Kán15]



(e) Light probe image. [JNVL10]



(f) Virtual representation. [CWW11]

(g) Spherical representation. [MD06]

Figure 2.1 – Different possible representations of environment maps.

positions, taking into account light color, reflection, object occlusion, or object shadows near the inserted object. Without this knowledge, the object denotes in comparison with the rest of the scene, and the result is of lower quality.

In this context, the scene can be divided into two parts: the **local scene**, which has the focus of a camera and contains the region of interest, and a **global scene** (or environment). In the various work presented here, different terms can be used to define 360° images: omnidirectional, spherical or panoramic image, panorama, wide or large angle image. In this thesis report, we will prefer the broader term of **environment map** to define them. Because an environment map is a 360° image, it contains the complete background information of the scene. This information can be various: landscape, light, or full geometry information. A panorama can be captured as an high dynamic range (HDR) imaging. An HDR image [RHD+10] is an image which presents a large range of intensities and thus, can serve as an environment lighting representation. Such an image contains information in both underexposed (dark) and overexposed (bright) areas.

First, in section 2.1.2, we define a panorama and explain in which way a panorama is linked to the concept of environment map. In the next section (2.1.3), we present different ways for environment maps acquisition. In section 2.5, we discuss the different utilizations of environment maps. Finally, in section 2.6, we review state of the art methods designed to orientate environment map.

2.1.2 Panorama of images

2.1.2.1 Mosaicing and stitching of images

It is possible to create a panoramic image from a set of images of the same scene. Each image has to present a recovery area (*i.e.*, overlapping) with the neighbor images. Finding similarity in overlapping areas permits to combine them to generate the panoramic image.

There are different ways to find similar key points in different images. For example, SIFT (Scale-invariant feature transform, [Low99]) or SURF (Speeded Up Robust Features, [BTVG06]) algorithms, computes descriptors (or visual features), on relevant pixels in the image. A descriptor is a vector of values containing information such the positions, scale, or orientation of the relevant pixels.
By comparing theses descriptors in the different images, it is possible to link relevant pixels of an image with suitable pixels of another one. The last step computes the homographic transformation to apply it and corrects the different projections of the two images.

Figure 2.2 – Panorama of created using a set of images. Courtesy of [BL07].

Other work, by Szeliski *et al.* [SS97], apply a set of transformations to keep consistency in the output mosaic.
They presents a way to construct cylindrical panorama using a sequence of images using cylindrical or spherical coordinates, and general perspective transforms. They operate a "gap closing" technique to fill the panorama and the potential holes due to rotation estimation. They finally present an algorithm to convert the mosaic into an environment map. In [Sze96], Szeliski *et al.* create high-resolution images using different frames of a video and align them to obtain a mosaic. They obtain different styles of mosaic, high-resolution images, panorama, and recover partial depth of the scene.

2.1.2.2 From the panorama to the environment map

There is no real difference between the different representations of panorama or environment map. In the following, we present different representations of environment maps which show structural differences likes image representation, better visual perception or hard disk size, but only the usage done with the environment map can reveal which representation is most adapted. For these reasons, there is no difference between panorama presented here and environment map as described in the next sections.

2.1.3 Environment map acquisition devices

Several devices exist to obtain an environment map as chrome sphere or the stitching of panoramas using a set of images and software which finds the correspondences between areas to obtain a unique 360° image. The use of a fish-eyes lens which gives images with a larger angle of view can reduce the number of needed images. A single image can be taken if the camera is oriented in the direction of the zenith. With this, we obtain a good representation of the incoming light. Indeed, most of the time, the relevant incoming light is positioned on the upper hemisphere. A 180° representation of the scene can be sufficient to represent the incoming light.

Another way to directly obtain an environment map is to use a dedicated camera. Most of the time, this device uses fish-eye lenses and are already calibrated. The 360° camera recreates the spherical image automatically.

For some applications which rely on virtual data, like videogame, it is possible to create a virtual environment map using a fonctionality of a software (as PBRT [PWH] or Povray [Pov]). Indeed, such software allows the user to create virtual scenes and more particularly to parametrize the virtual camera. In addition to usual parameters as position, direction or up vector, the parameters of a camera can be set as: a very large angle to capture spherical images.

2.1.3.1 Reflective sphere

A way to create an environment map is to use a reflective sphere placed in the center of the scene which reflects the scene at 360° as a mirror (see figure 2.3). Taking such a picture is a way to capture the entire scene. To obtain a good quality of the reflected illumination, the sphere has to be placed close to the camera. However, the capture device camera is also reflected in the reflective ball.

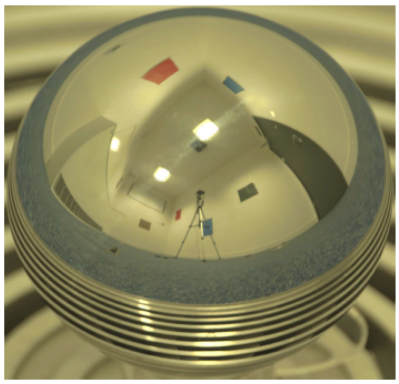


Figure 2.3 – Chrome sphere captured. Courtesy of [JNVL10].

The photographer can remotely control the camera to avoid to be reflected in the reflective sphere. Indeed, as the capture device, it is not desirable to have the cameraman in the image. It is possible to virtually remove the reflection of the camera. For example, the software HDRshop [4] allows, from two pictures taken at 90 degrees of the same mirror ball, to define two masks excluding the camera in each image, and to recreate a single image of the scene presenting an environment without the camera [5].

[4] HDRshop website
[5] See the HDRshop website, section "tutorials", "Creating a Light Probe Image" to more details.

Unger *et al.* [UGY07] use three cameras (Red, Green, and Blue channels) to capture chrome sphere images. This capture allows to obtain a high dynamic range (HDR) image, and thus use it to perform image-based lighting. Different uses are possible using a chrome sphere. Corsini *et al.* [CCC08] acquire spatially-varying lighting environments in HDR from photographs of two reflective spheres. They also recover the positions of light sources, which they use for a coherent rendering of virtual objects.

Chajdas *et al.* [CWW11] automatically compute important locations to guide light probe positioning in a virtual space. An application to video games to help the game designer to obtain environment maps is shown. Jacob *et al.* [Jac07] and later [JNVL10] propose a method to calibrate light probe images. An application of the radiance capture to inverse lighting and relighting of scenes is finally shown.

2.1.3.2 Camera 360

Various 360° cameras have been developed last year, all based on the same principle. Several fish-eyes (large angle) lens are placed all around the camera to cover the total sphere around. Because a fish-eye lens is not a 360-degrees lens (but maximum 200 degrees), several lenses are needed to obtain a complete circle.

When a picture is taken, all the lenses are triggered at the same time. That allows reconstructing a single 360-degree image which is coherent in term of lighting. Indeed, if the lenses are triggered at different times, the illumination can change thus leading to artifacts in the final pictures.
Because the lens positions relative to each other are known, the device reconstructs a 360-degree picture automatically. Some artefacts can also appear if the lenses are exposed in a different way to the light. Indeed, if a lens is more exposed than others, a delimitation can be seen after the reconstruction step.

Gledhill *et al.* [GTTC03] review different panoramic image capturing systems. They present different devices to the capture, and ways to obtain omnidirectional images. The survey also highlights different problems linked to the recovery of such images such as distortion, or a lack in the dynamic range caption, and criteria link to the spherical images.

Devices with a large field-of-view lens (see figure 2.4, top line) allows to obtain spherical images. In [HCP07], Havlena *et al.* use a 180° fish-eye lens to capture video sequences and

[5]Photo from wikipedia.

Figure 2.4 – Top: a fish-eye lens mounted on a camera. Bottom: Different 360° cameras (from left to right): Ladybug, Bublcam and Ricoh Theta.

obtain a 3D reconstruction of a city. Later, Torii, Havlena and Pajdla [THP11] propose a pipeline for trajectory estimation, and image stabilization of omnidirectional images. A fish-eye lens of 183° view angle is used in this work to obtain high-quality images.

Lhuillier *et al.* [LN15] build a DIY ("do it yourself") helmet-mounted with four GoPro cameras to obtain 360° images in the horizontal plane. They synchronize and calibrate the four cameras by computing time offsets between cameras and inter-camera rotations.

More recently, dedicated devices to obtain such images have been proposed on the market. The simplicity of use and the fact that the synchronisation and calibration steps are already done, make of these cameras practical tools and allow easier or new uses. Sato *et al.* [SY10] use a Ladybug camera (see figure 2.4, bottom left), placed at different viewpoint in an environment to obtain a 3D environment using pairs of high-resolution spherical images. Kim *et al.* [KH13a] use a rotating line scan camera to captures the full surrounding visible scene.

Bodington *et al.* [HBT15], use a Ricoh Theta camera (see figure 2.4, bottom right) for 360° rendering for virtual reality applications. Later, Ran *et al.* [RZZY17], uses a spherical camera mounted on a wheeled robot to obtain 360 fish-eye panorama datasets and allow the robot navigation. The robot and camera device are shown in figure 2.22. Akbarzadeh *et al.* [AFM+06] propose a device mounted on a car using three cameras to create a horizontal

field of view of 120 degrees. Another camera is tilted upward to obtain a vertical field of view. The aim is not to obtain a complete 360 image recovery but the idea of a large field of view and a pointed up camera is similar to omnidirectional images devices.

2.1.3.3 Virtual environment map

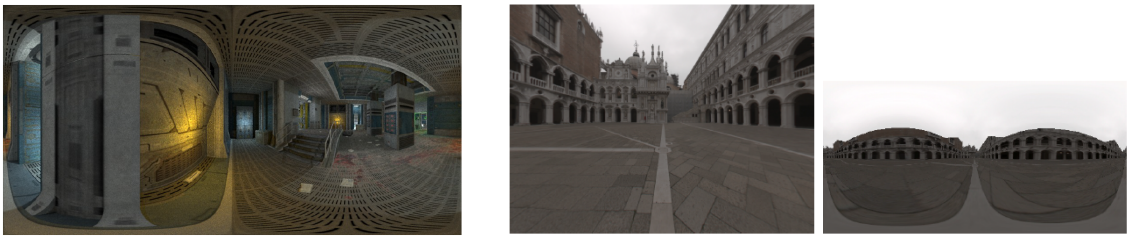


Figure 2.5 – Left: virtual spherical environment map, courtesy of [CWW11]. Right and middle: synthesized environment map (right) from a single image (picture in the middle), courtesy of [LE10].

In some cases, it is not possible to obtain an environment map if, for example, it is far away, inaccessible, or unreal. Moreover, the input data can come from unknown sources (internet, database). Using the hypothesis that only an approximation of the environment is necessary to simulate a coherent illumination, Lalonde *et al.* [LE10] (see figure 2.5), synthesize a possible environment map using a single image. Other more original works generate environment maps from the eye reflection [NN04]. Kán [Kán15] generate an environment map using a mobile phone.

2.1.3.4 Analytic model

A sky-dome can be analytically modeled [PSS99] [HW12]. The idea is to take into account physical parameters (for example, light source positions, ground albedo, and material, luminosity or turbidity) and to compute a correct sky image corresponding to reality (generally a spherical image) which can be used as environment map latter. Different results are shown in figure 2.6. This kind of acquisition aims to overcome limitations associated with capture time (necessarily low for navigation in an environment for example), memory cost or easily edit of sky parameters (useful in many fields as Fx or rendering). Wilkie *et al.* [WH13] predict the appearance of planet skies with different sun numbers, size, and color, different atmosphere composition (and other parameters). More recently, Satylmys *et al.* [SBRCD17] trained a memory and time efficient machine learning model and proposed weights for recreating skies' models already existing: Preetham, and Hosek & Wilkie.

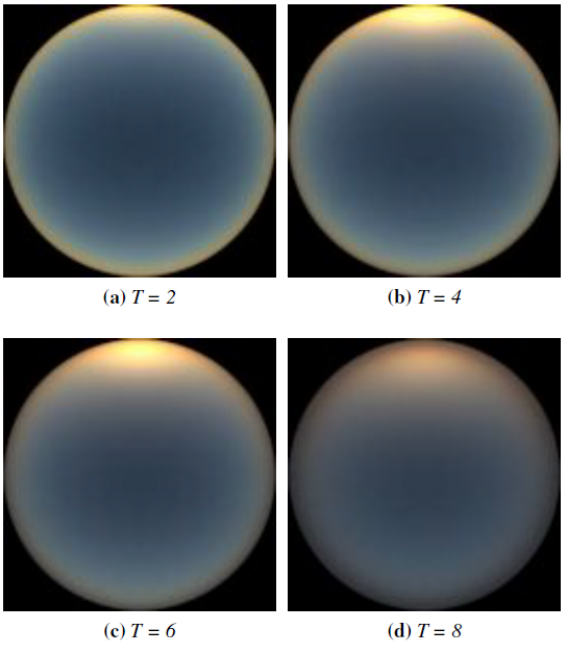


Figure 2.6 – Result of sky-dome modeling using analytic model. The figure show several modeling of the sky using different turbidity values with a 4° solar elevation. Courtesy of [HW12].

2.1.4 Conclusion

We saw in this part environment map acquisition approaches. These approaches provide more or less the same information, but the usage done with the result of the acquisition will determine the best way to use. Indeed, for the example of a virtual visit of a museum (see section 2.5.6) a camera 360 (see section 2.1.3.2) will be an easier way to obtain panorama with real detail than to use a virtual environment created with a computer (see section 2.1.3.3).

In another way, using an analytic process (see section 2.1.3.4) to create an environment map for a relighting usage, will provide more flexibilities to change the light parameters than using a chrome sphere device (see section 2.1.3.1) and take a large set of photography to obtain a diversity of panorama.

2.2 Image mapping

Different representations of the environment maps exist depending on their use. In this section, we target representations for real-time video game-oriented rendering. We identified several mappings: cubic, spherical, and paraboloid. For each representation, we study specificity and aliasing results. Also, we consider that it is necessary to have a visual result

that is as uniform as possible, avoiding distortions, artifacts and, aliasing, while having the smallest memory footprint and being adapted to the interpolation mechanism of GPUs, in this case, linear interpolations.

2.2.1 Dual paraboloid mapping

Another representation is based on paraboloid maps. The method uses two textures to represent the reflection on two hemispheres represented by paraboloids [HS99]. This approach has the advantage of being independent of the point of view and that only two textures are needed to cover all directions, which allows them to be generated on GPU in only two rendering passes. However, access to the texture requires reprojection, to take into account the parabolic surface and the deformation of the paraboloids complicates the interpolation in this type of representation. The increase in available GPU memory and power has made the use of parabolic maps obsolete.

2.2.2 Cube mapping

Nowadays, the representation used on GPUs for environment mapping is a cube. Indeed, the representation in the form of a cube allows rendering with a geometric cost limited to the rasterization of twelve triangles which is much simpler than the rendering of a sphere. With today GPUs' power, this consideration may be less of a priority, but it should not be forgotten that until a few years ago, managing geometric complexity was a real challenge and the amount of available graphics memory was limited. Cube mapping, designed by Greene [Gre86], has been supported since the early 2000s with the introduction of cubic textures. The ideal representation, in this case, is to store the six images corresponding to the six faces of the cube. Several approaches are used:

- six images;
- one map with a row of six faces;
- one image of an unfolded cube where 50% of the image is empty and wasted.

In the latter two cases, the visual interpretation of the texture is not easy.

2.2.3 Equirectangular mapping

Another solution is to store the spherical environment map by projecting it onto a single rectangular image. That is a problem widely known to geographers, for which there are dozens of types of projections that offer different compromises between shape and area

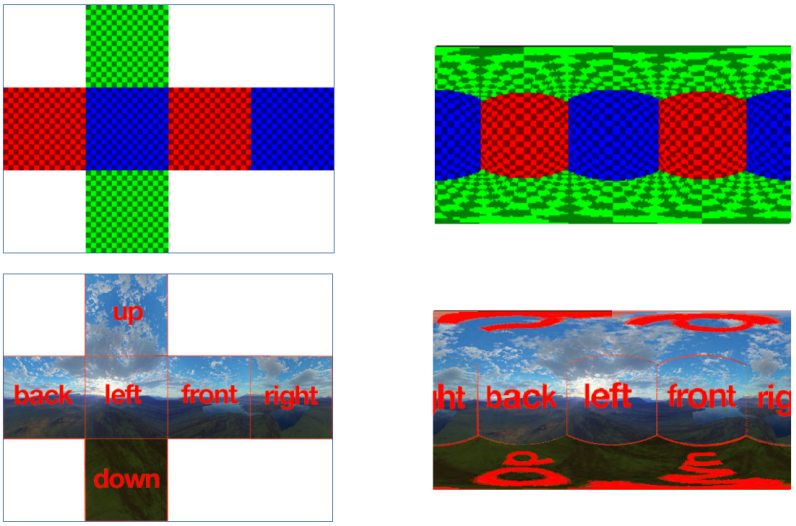


Figure 2.7 – Left: A six-face unfolded cube representation. Right: An equirectangular projection of the same environment: about 50% of the pixels represent the upper and lower sides while the other four sides share the rest of the area.

distortions as discused in [DLR16]. The solution commonly used for environment and panorama maps is an equirectangular projection. It has the advantage of using the entire available surface area while being easy to visualize. However, it introduces deformations at the poles of the sphere where, for example, the first line of pixels is projected on the same point of the sphere and the information density is low at the horizon (see figures 2.7). When used in a GPU cube mapping, reprojection on all six sides of the cube introduces deformations that generate interpolation errors. As illustrated in figure 2.8, this very low-resolution environment map (64×64 pixels per face) highlights the deformations, on the left-hand side, one of the vertical faces, and on the right-hand side, on a face that corresponds to a pole of the sphere. Interpolation errors are due to bilinear sampling of texture access on GPUs in areas where correct interpolation should be calculated more on an arc than on a square. It is necessary to use higher resolutions than the unfolded cube texture format to achieve a similar qualitative result to limit these disadvantages.

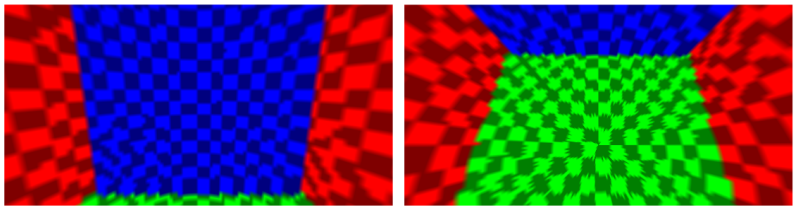


Figure 2.8 – Equirectangular projection to cube map aliasing and distortion.

Isidoro [IM05] pointed out in 2005 a weakness in the filtering of cube mappings with GPUs: the interpolation is false when approaching the borders of the cube faces (see figure

2.9), in fact, the color of the neighboring face is not taken into account. Thus, in some cases, artifacts may appear at the junction of the faces of the cube. An interpolation taking into account the neighboring faces avoids this problem: graphic APIs have adopted this solution since 2009.

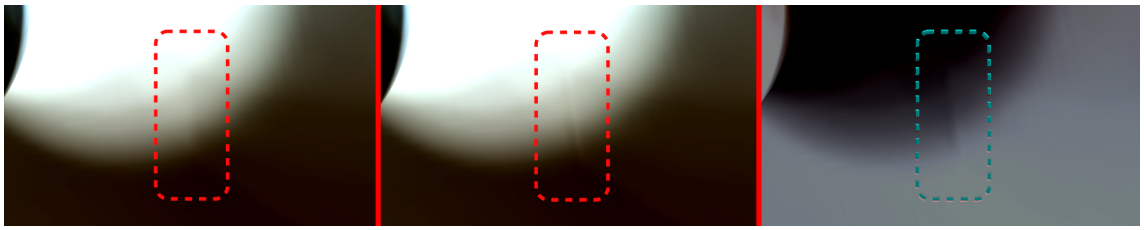


Figure 2.9 – Left: Seamless artifact. Middle: seam artifact. Right: Difference between images. Contrast have been enhanced.

2.2.4 Discussion

Cube mapping is the technique that allows a much better distribution of the sample than sphere mapping or dual paraboloid mapping. There is an even better distribution proposed by Wan [WWL07] which consists of using the shape of an iso-cube that remains compatible with GPU implementations of cubic textures. As illustrated in table 2.1, the representation of the environment map in the form of a cube present most advantages and, therefore, is the most used solution in real-time rendering. We can notice the over-representation of the z-direction with the use of rectangular equiprojection texture, which explains the need for a higher resolution than for the unfolded cube texture to obtain the same visual quality.

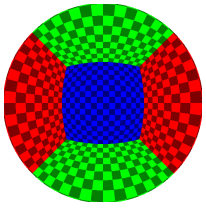
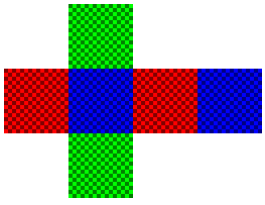
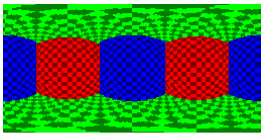
	Dual Paraboloid	Cube	
			
% used in texture	78	50 or 100	100
% used in memory	78	100	100
Interpolation quality	low	very good	varies
Pixel direction distribution	less in y	balanced	50% in z
Distortion	/	none	on poles
Sampling	bad	good	
Use	deprecated	common	

Table 2.1 – Comparison of texture representation.

2.3 Background on rendering and light reflection

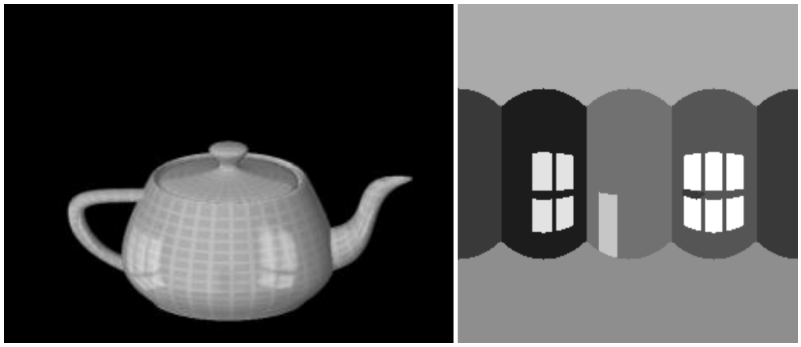


Figure 2.10 – First environment reflection. Courtesy of [BN76].

The first use of environment maps, introduced by Blinn in 1976 [BN76], was the reflection mapping to simulate the reflectivity of a surface by interrogating the environment map according to the direction of a ray reflected by a surface (see figure 2.10). Greene [Gre86] extended the scope of environment maps by using a cube to represent the environment map, then called *cube mapping*. This choice strongly influenced the history of rendering on GPUs since this form of cubic texture is implemented directly in the various graphics APIs and is still used. He uses this environment map to represent the background of a scene, introducing *sky boxing* for lighting simulation. Greene [Gre86] takes up the concept of reflection mapping, in the context of offline rendering from the idea of Whitted [Whi79]: a pixel is not considered as a simple point but as a surface. For the reflection, it is then necessary to consider the portion of the environment map covered by the back projection of a pixel. This poses the challenge of sampling the environment map, as the size of the area covered may be large under certain conditions (see figure 2.11). Fortunately, the use of pyramidal texture or mip-mapping [Wil83] limits the number of texture accesses and allows to remain usable at an acceptable cost.

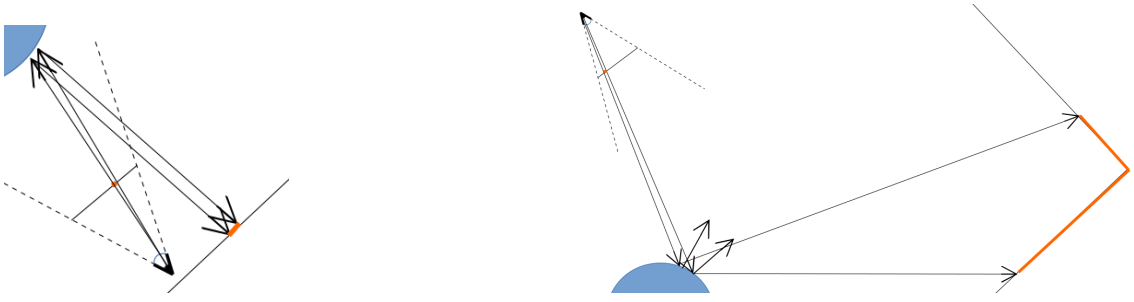


Figure 2.11 – Pixel area reflection. Left: A pixel with a small back-projection on the environment map. Right: The pixel footprint covers multiple faces of the cube map.

The use of sky-boxes is widespread for rendering applications, especially in the field of

real-time rendering on GPUs. It is a question of a global environment map: all the pixels on which no geometry of the scene is rasterized will hang the colour corresponding to the direction of view in the sky-box. This technique has been implemented on GPUs since the early 2000s.

Similarly, the reflection and refraction effects can be simulated by computing the direction of a reflected or transmitted beam as a function of the normal to the surface and the view direction. It should be noted that refraction, in the context of rasterization rendering, is more a visual effect than real refraction since the direction of the beam is only modified at the point of entry of an object. The sky-box allows adding realism for a cost limited to a single access texture per pixel.

However, occlusions by objects of the scene are not taken into account. It is possible to partially correct this inconvenience by using dynamic sky-boxes updated at each image allowing, for example, to have the reflection of an object of the scene on another one (see figure 2.12, left image). It can be noted that these reflections are only accurate for a distant environment, theoretically placed at infinity, since the result is linked to a direction regardless of the position of an object in the scene. Whenever close objects are represented, parallax artifacts are produced.

Figure 2.12 – Example of sky box with dynamic reflection. Left: on a sphere. Right: on a transparent statue with refraction.

2.4 Image based lighting

The purpose of using environment maps is, on the one hand, to reduce calculation costs by limiting the size and complexity of the scene to be modeled and rendered, and, on the other hand, to integrate real-world data to increase the realism of rendering. A significant part of the research aims to make the best use of environment maps as a light source to illuminate a scene. These so-called image-based illumination methods will make it

possible to simulate lighting of materials with increasingly realistic bidirectional reflectance distribution functions or BRDFs. Reflection mapping corresponds to the case when the incoming radiance of the environment map depends only on the direction.

2.4.1 Introducing BRDF

First of all, it is necessary to recall the rendering equation (2.1) for a non-emitting surface that allows the radiance to be calculated in any direction (see figure 2.13). Reflectance can be defined as a spherical convolution of the incident illumination and the BRDF.

$$L_o(p, \omega_o) = \int_{\Omega} f_r(p, \omega_i, \omega_o) L_i(p, \omega_i) n \cdot \omega_i d\omega_i \tag{2.1}$$

- $L_o(p, \omega_o)$ is the radiance from point p in the direction ω_o;
- $f_r(p, \omega_i, \omega_o)$ is the BRDF;
- $L_i(p, \omega_i)$ is the incoming radiance at point p from direction ω_i;
- n is the normal of the surface at point p;
- $\int_{\Omega} ..d\omega_i$ is the integral of Ω hemisphere;
- $n \cdot \omega_i$ is the attenuation factor related to the angle of incidence.

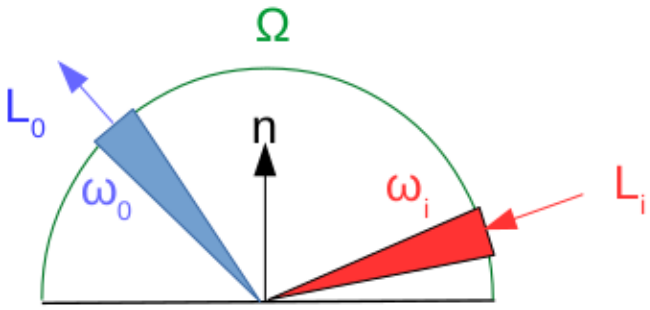


Figure 2.13 – Reflectance scheme.

A BRDF is generally composed of diffusion and reflection. Figure 2.14 illustrates some basic BRDF functions. By combining basic BRDFs, it is quite easy to describe models of materials used in image synthesis. That allows the appearance of materials to be calculated with isotropic BRDF (one can turn around normal without changing the reflection).

Diffusion results from Lambert's law where the surface of material also appears diffuse from all directions. This BRDF f_{rd} is therefore a function of the diffuse coefficient, that is

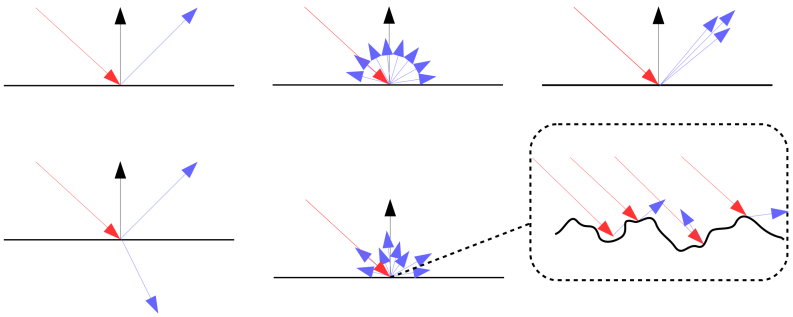


Figure 2.14 – Several BRDF functions. Top, left: the BRDF of a perfect mirror. Top, middle:pPerfect diffusion, according to Lambert's law. Top, right: diffuse reflections. Bottom, left: transmission reflection, when part of the light passes through the material according to the laws of Snell-Descartes. Bottom, middle: material modeling based on micro facets like Cook-Torrance's model. Bottom, right: zoom on the surface to illustrate the roughness of the material with reflections in various directions and self-occlusion resulting from the micro-facet model.

the diffuse albedo ρ_d of the material (see equation (2.2)).

$$f_{rd}(p, \omega_i, \omega_o) = \frac{\rho_d}{\pi} \tag{2.2}$$

In the case of a perfect mirror or specular reflection, BRDF is defined in equation (2.4) by the reflection ray r defined in equation (2.3), and depends on the reflected direction ω_o, the surface normal n and the reflective coefficient k_s .

$$r(n, \omega_o) = 2(n \cdot \omega_o) \times n - \omega_o \tag{2.3}$$

$$f_r(\omega_o, n) = k_s \times r(n, \omega_o)\cdot \tag{2.4}$$

In computer graphics, Phong is one of the most widely used reflection models, which makes it possible to represent diffuse materials, like plastics, by combining an ambient part simulating a fake global illumination, a diffuse part according to Lambert's model and an additional reflection lobe. This model does not correspond to any real physical model and admits some approximation: the size of the diffuse reflection lobe is the same in all directions. The roughness of the material that influences the size of the reflection lobe is a function of N, the specular power index (the higher N is, the closer we get to the behaviour of a mirror while when N is low, the closer we get to a diffuse reflection). The importance of the specular lobe in the Phong model depends on the reflective albedo k_s.

$$f_r(\omega_o, \omega_i, n) = k_s \times \frac{(r(n, \omega_i) \cdot \omega_o)^N}{n \cdot \omega_i} \tag{2.5}$$

For some materials, albedo is not a constant and varies according to the angle of incidence of the view direction, for example, in water: the sea at the horizon reflects the sky while seen from above a viewer can see the sea beds. These types of materials comply with the Fresnel law for reflection and transmission in the case of translucent materials. Its formulation being complex, the Schlick approximation (equations (2.6) and (2.7)) is often used, where n_1 and n_2 are the refraction indices of two different media. In a typically rendering example, the first medium will be air ($n_1 = 1$) and the second will be the surface material whose appearance is to be calculated. θ is the angle between the normal and viewing direction.

$$k_s(\theta) = R_0 + (1 - R_0)(1 - cos\theta)^5 \tag{2.6}$$

$$R_0 = (\frac{n_1 - n_2}{n_1 + n_2})^2 \tag{2.7}$$

There remains a more sophisticated type of BRDF functions, for anisotropic materials, which is characterized by a reflection whose direction is dependent on the direction of incidence, unlike isotropic materials. This type of BRDF allows, for example, to simulate the appearance of brushed metal (Lafortune's model [LFTG97], which is an improved Phong model). There can be several reflection lobes whose directions can be variable. That allows more complex materials to be represented with retro-reflections. One of the physical models of BRDF now used in real-time physics-based rendering is the Cook-Torrance model. In this model (see equation (2.8)), there is a diffuse part but also a variable specular part based on the estimation of the roughness of the material as a function of a microfacet distribution function D representing the surface of the material, the Fresnel coefficient F and a function G which statistically describes the geometric effects of occlusion of the microfacets.

$$L_o(p, \omega_o) = \int_\Omega (k_d \frac{c}{\pi} + k_s \frac{DFG}{4(\omega_o \cdot n)(\omega_i \cdot n)}) L_i(p, \omega_i) n \cdot \omega_i d\omega_i \tag{2.8}$$

2.4.2 Hardware consideration

To fully understand the choices of research axes, it is necessary to take into account the evolution of hardware constraints related to the GPU and graphics APIs: nowadays it is not uncommon to use a large number of rendering passes and to combine the use of numerous textures, just as in the past the limited resources of GPUs could justify technical choices that have for some completely disappeared, such as sphere mapping. For example, methods based on parabolic environment maps were quite popular in the early 2000s: it had the

advantage of using only two textures to cover all directions and could be generated in two rendering passes where the cost was three times higher with a cubic representation knowing that the cubic representation was not supported natively by all GPUs. With the hardware support of generalized cube mapping over many years, the increasing amount of graphics memory and the computing power of GPUs, these methods now have less value.

2.4.3 Diffuse lighting

The problem of illumination by an environment map is well illustrated with the case of diffuse illumination: to compute lighting in one direction, it is simply required to sample the half-space of the environment map around this direction by taking into account the cosine weight of the angle between the direction of the sample (see equation (2.2)). The problem is similar for specular reflections as they require to sample the portion of the environment map covered by the specular lobe. For diffuse illumination, Greene [Gre86] uses a brute force approach by pre-computing for a cubic environment map the corresponding irradiance map (see figure 2.15) or for each direction the diffuse contribution is stored. This solution is extremely costly in terms of computation time: at the beginning of the 2000s, computing an irradiance map of $64 \times 64 \times 6$ from an environment map of $300 \times 300 \times 6$ took about 2 hours. This significant cost is easily explained by the complexity of the problem to be solved: the complexity of the computation of an irradiance map of resolution i from an environment map of resolution e is in $\mathcal{O}(e^2 \times i^2)$.

Figure 2.15 – Environment map and the corresponding irradiance map.

Fortunately, irradiance maps representing a low-frequency signal (a convolution on a hemisphere can be compared to a low-pass filter), do not require a high resolution. The advantage of this approach in rendering is that diffuse lighting is single texture access. The use of irradiance maps works equally well for cube mapping as for paraboloid mapping [HS99]. Debevec *et al.* [Deb98] use the same principle with sphere mapping. However, sphere mapping is a view-dependent representation. In Cabral *et al.* [CON99], a non-linear interpolation method between these maps allows to cover all directions in real-time

rendering on GPU from several irradiance maps of the same environment from several angles.

To avoid the costly environment map pre-filtering, Ramamoorthi *et al.* [RH01] point out that the generation of irradiance maps results in images with low-frequency signals, so we can allow a first rendering approximation which consists in computing the diffuse radiance by vertex and then, interpolating afterward by fragment while keeping a qualitative result. In addition to its computation time, the irradiance map involves storing an additional texture, and depending on the GPUs' capabilities, to add an additional rendering pass which is detrimental to performance. To avoid these disadvantages, the calculation of the diffuse component of a material is done by filtering the environment map in the frequency domain based on spherical harmonics (equivalent to a Fourier transform on a sphere). These are spatial functions whose weighted sum makes it possible to approximate a function on the unit sphere such as irradiance. Ramamoorthi *et al.* show that a combination of spherical harmonics of order 0 to 2, *i.e.*, nine functions, is sufficient to obtain an approximation close to one percent of the convolution result of a hemisphere on the environment map. The first step of pre-filtering the environment map makes it possible to obtain the nine RGB triplets that serve as weights when adding the contributions of the different harmonic functions. This pre-filtering is fast (complexity: $\mathcal{O}(e^2)$), in the order of a second, and makes it possible to obtain a much more compact representation than an irradiance map, even of low resolution. During the rendering phase, the computation of the diffuse colour is a simple polynomial sum from the normal and the coefficients calculated during a pre-filtering step. King [Kin05] and Kautz *et al.* [KDS04], inspired by the spherical harmonics approach, calculate in real-time the irradiance of a dynamic environment map on GPU.

2.4.4 Glossy reflection

One extension of the environment maps is its use as a source of reflection with the case of glossy reflections (see figure 2.14, top right). The approach is similar to the one used to calculate irradiance maps except that the convolution of the environment map is not performed on a hemisphere; it is restricted to the specular lobe. Heidrich *et al.* [HS99] propose to pre-filter the environment map to make a new texture which contains, for each direction, the outgoing radiance corresponding to the specular reflection lobe of a Phong model. This new texture is known as a specular irradiance environment map. In addition, a lookup table with pre-calculated Fresnel factors is used in order to represent more realistic materials at lower calculation costs. A disadvantage of this type of filtering is

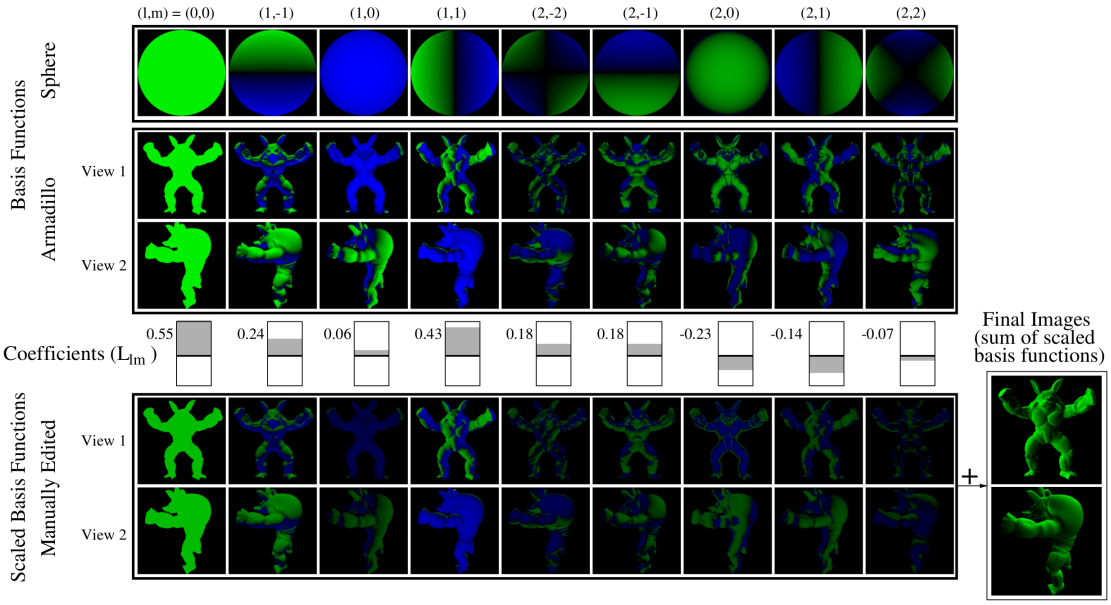


Figure 2.16 – Spherical harmonic decomposition (courtesy of [RH01]) Top: on the first row, the nine spherical harmonic functions with the green and blue colours which represent respectively the positive and negative values displayed on a sphere. Bottom: example of reconstruction on a colour channel from the different harmonics weighted by the associated coefficients L_m.

that it is by nature related to a single specular power, so having different Phong materials in the same scene implies having a texture per type of specular lobe. This first filtering is adapted to the Phong model which is only an empirical model, with BRDFs of more realistic material, the shape of the reflection lobe varies according to the incident angle of the viewing direction: the lobe becomes narrower and longer for grazing angles. Kautz *et al.* [KM00] propose to filter the environment map with several specular lobes to obtain reflection maps for each of them and to be able to make more complex BRDF. Hierarchical filtering is applied using the mipmapping, which allows fast but lower quality filtering. Based on Lafortune's DRDF, Macilster *et al.* [MLH02] filter the maps of the environment according to different specular exponents (stored in the different MIP maps levels of the glossy reflection map) to be able to render materials notoriously anisotropic. That is performed by adding a texture to store the anisotropy direction of the material and the specular exponent which allows to determine between which MIP map levels to interpolate in order to obtain an approximation of the glossy reflection. Kautz *et al.* [KDS04] use an identical approach based on the MIP map, which does not require any pre-filtering even if the result is approximated. Hensley *et al.* [HSSL05] propose to store the environment map in a summed-area table which has the advantage of being able to quickly add a rectangular portion of an image with a constant reading number whatever the size considered. All these

techniques allow obtaining a convincing rendering of various types of BRDFs without taking shading into account.

2.4.5 Global illumination

The illumination methods presented above consist of a convolution of the environment map by a BRDF. Sloan *et al.* [SKS02] decide to add element, visibility, by limiting to the basic lighting frequency environment. Thus, this technique adds visibility to the principle of Ramamoorthi *et al.* [RH02]. In this case, it is the rendered-object self-shading which is encoded as spherical harmonics in a pre-step. This technique, known as the pre-calculated transfer function, can be used to simulate the first and second inter-reflection rebound, glossy reflection and subsurface scattering materials. This approach is limited to non-distorting objects. In 2005, Sloan *et al.* [SLS05] present an extension to take into account animated objects based on particular classes of spherical harmonics, the zonal harmonics, whose particularity is to be invariant for a specific axis. Thanks to this specificity, rotations are easier to compute than with classical spherical harmonics. That permits a real-time rendering of animated objects while allowing the management of bumpy surfaces with wrinkles. Starting from the observation that spherical harmonics are not very efficient to represent high-frequency signals even if their number is significantly increased, Ng *et al.* [NRH03] propose to filter the environment map with wavelet functions that have the advantage of representing both high and low-frequency signals. Thus, it is possible to obtain hard and realistic shadows with a decomposition of 200 wavelets to obtain a qualitatively close rendering of 20 spherical harmonics decomposition while remaining in interactive time.

2.4.6 Sampling

A classic approach to solving the integration of the rendering equation is to use the Monte Carlo method which, to solve complex mathematical problems, is based on a probabilistic approach (see equation (2.9)). When using an environment map as a light-map, the rendering equation is approximated by sampling the environment map to cumulate the N contributions of each sample depending on its visibility $V()$ weighted by the importance sampling function commonly called *pdf*. Importance sampling is a method for reducing variance in Monte Carlo integration.

$$\int_{\Omega} f(x)dx \approx \frac{1}{N}\sum_{i=0}^{N}\frac{f(x_i)}{pdf(x_i))} \tag{2.9}$$

$$L_o(\omega_o, n) \approx \frac{1}{N} \sum_{i=0}^{N} \frac{f_r(n, \omega_i, \omega_o) L_i(\omega_i) V(\omega) n \cdot \omega_i d\omega_i}{pdf(\omega_i))} \tag{2.10}$$

This gives equation 2.10 in the case of purely directional lighting with an environment map. The sampling methods proposed by the community can be classified by the choice of the weighting function; one of the first choices is to consider that the sampling importance function should be the inverse of the BRDF. This approach is particularly well suited to very specular BRDFs but loses effectiveness for diffuse BRDFs. Several works are oriented towards this technique, including Lafortune *et al.* [LFTG97]. A second solution is to choose the distribution of the incident radiance as *pdf*. Agarwal *et al.* [ARBJ03] propose a hierarchical representation of the environment map in order to create more samples in the areas with the most important radiance values. Thus, with only 300 samples, the result is much better than with 1,000 purely random samples while having convincing results with no noise, even with a small number of samples. However, this type of technique is more adapted for diffuse than specular BRDFs. To improve the two previous sampling families, the idea is to combine both types of techniques for the sampling importance function to be based on the entire integral and not on a single component, in the form of a product of sampling importance like [CJAMJ05], using wavelet to generate the samples, and [JCJ09], based on spherical harmonics filtering.

Gabal *et al.* [CMS87], in the context of offline rendering, consider the object surfaces as bumpy to generate specular reflections. To achieve this, they consider surfaces composed of triangular micro facets distributed on a regular grid and generate a pre-computed table in four dimensions taking into account partial or complete occlusions of the faces to represent the BRDF in the form of spherical harmonics. This method presents advantages for very rough surfaces, as the number of spherical harmonics required for a good approximation is low.

2.5 Environment map use

2.5.1 Introduction

Environment map images are often used for lighting or relighting a scene. That has been introduced early, as for example with Daubert *et al.* [DSSD97] who used a sky-dome to compute the global illumination of an outdoor scene. Pellacini *et al.* [Pel10], [BCD+13] develop an interface which allows the users to edit the lighting effects of an HDR environment map. The use of panoramas as input data for an illumination algorithm is more and

more common because of the additional information that these bring. More, we can note there that most methods described in this section do not use HDR environment map.

There exist methods to compute a 3D model of an outside environment using panoramas, [MK09] [LN15]. The panoramas are acquired in different ways either from a set of 5 GoPro cameras mounted on a helmet used by a user, or from Google Street View [6] panoramas. Similary, panorama can be used to reconstruct 3D objects, such as building or small objects [LSD+14] [KH15]. Other methods recover the trajectory of stereoscopic cameras, as in [THP11], using similarity between panoramas. Other methods [MD03] [MD06] [VH13] recover the rotation of a 360° camera or to realign panoramas. Other usages of panoramas are possible like with [HBT15] where 360° images are used for virtual reality application or in [DCL11] where a method which finds correspondence between panoramas is shown.

Cinema, specialy in special effects, was an early field to use environment mapping. We can mention the "pseudopod" creature of *The Abyss* (1989, James Cameron), or the "T-1000" character of *Terminator 2: Judgment Day* (1991, James Cameron) for which the company *Industrial Light & Magic* use reflexion mapping (see, [R+09], section 3.1 or Paul Debevec website, [Deb06]). More recently, the same kind of technique has been used by *Marvel Entertainment* to recreate Iron man armor in the movie *Iron man 2* (2010, Jon Favreau, see [Sno10] for several examples).

2.5.2 3D from environment map

Several methods have been developed these last years to obtain 3D information from panoramas or environment maps, obtained with different devices. Some of them, use a 360° camera [TITO15] [MK09], and others camera and a GPS [AKRS11] or a camera with a fish-eye lens [KH13b]. Google Street View spherical images [THP09] [SPY11] [MK09] are used in several papers because of the size of the Google's database which present spherical images, taken with a unique device (at least for a given area), at regular distances (see figure 2.17). The video stream can also be used either to generate a large set of panoramas related to each other in a known order [LN15] [AKRS11]. On the opposite, the authors of [KHFH11] recreate a 3D environment using a single image and user annotations. Other methods use a set of spherical images [LSD+14], [KH13b], [KH14] or [TITO15].

Most of the methods presented in this section use structure-from-motion and feature extraction to generate output data. Structure-from-motion is a technique which allows

[6] Streetview website

Figure 2.17 – Input data: Google Street View panoramas. Courtesy of [THP09].

estimating three-dimensional structures using two-dimensional image sequences. This kind of technique is mainly used when the motion of the camera is known and are more or less the same between the different 2D-images. For these reasons, the methods using Google Street View panoramas ([THP09], [MK09]) are especially well adapted for using structure-from-motion. Others [SPY11] use epipolar geometry to generate a 3D model using statistical geometric constraint. Structure-from-motion is also used with video stream [LN15] [AKRS11]. Videos present the same characteristics. All have the same gap between images (the camera motion is uniform), which allow a simple estimation of the camera motion.

An important step of the 3D reconstruction is to register the different images. A common way is to use feature or element extraction as in figure 2.18. Most research employ SIFT or SURF for feature extraction ([THP09], [HHJ+15]). These methods compute descriptors (or visual features), on relevant pixels in the image. A descriptor is a vector of values containing information as the position of the relevant pixels, scale, or orientation. Because these descriptors are invariant (*i.e.*, they have the same values) to the translation, rotation, scale, and orientation, their value stays the same in different images. By comparison, it is easy to link a panorama (or a part of a panorama) with another one. Kim *et al.* [KH14] use SHOT and FPFH descriptors which they define as "best 3D descriptors" (defined in other studies). A year later, Kim and Hilton [KH15] use Hough transform to obtain a 3D representation of the scene. Finally, in [AKRS11], the panoramas are split into 64x64 tiles to extract features and estimate the pose of the panorama.

Several methods solve for 3D reconstruction based on points ([THP09], [TITO15]) or surfaces ([LN15], [HHJ+15], [MK09]). Kim and Hilton [KH13b] [KH15] propose two methods based on cubic projection and using facade alignment for the outside reconstruction

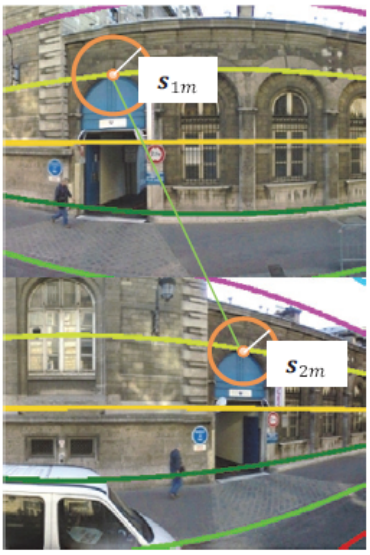


Figure 2.18 – Visualisation of features in panoramic images. Courtesy of [SPY11].

of streets and buildings. Both methods offer respectively a plane-based urban scene 3D reconstruction and a 3D bloc world reconstruction to estimate the surfaces of the objects (see figure 2.19).

2.5.3 360° video editing and blending

In order to create 360° images/videos, it is necessary to perform blending. Zhu *et al.* [ZLW+18] compares six different algorithms (feather, multi-band, modified Poisson, mean value coordinate, multi-spline, and convolution pyramid blending) in term of CPU and GPU memory utilisation and execution time. They use six different metrics (objective evaluation) for comparison. They also compare the result to a simple stitching algorithm. The study is made on several outdoor and indoor scenes (and a benchmark is thus proposed) to evaluate algorithm results in different environments. Moreover, a subjective evaluation is done by 33 users to test the visual quality of the different blending algorithm.

2.5.4 Augmented reality - Virtual reality

Augmented and virtual reality is a very active field of research. They beneficiate from the arrival of on-the-shelf devices (like virtual reality headsets) associated with development software (like ARCore (Google) or ARKit (Apple)). One challenge that remains in augmented reality is the addition of a virtual object in a real scene with correct common lighting. Many approaches were developed since the early ages of computer graphics

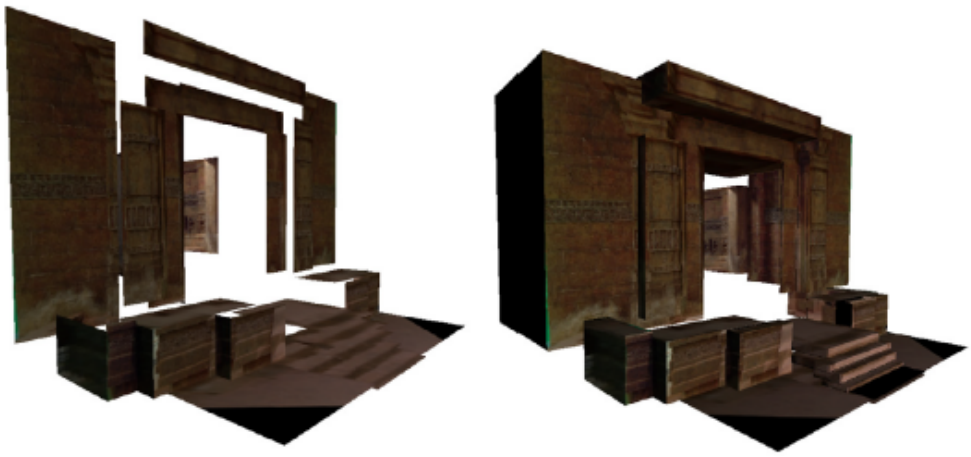


Figure 2.19 – Left plane reconstruction. Right: bloc reconstruction. Courtesy of [KH15].

[Jac07] [JL06] that consider coherent common lighting. Some of them use environment maps.

For example, Sato *et al.* [SSI99] propose a method to superimpose a virtual object onto a real scene. A radiance map is constructed using an omnidirectional stereo algorithm. Debevec [Deb08] demonstrates a method which uses scene radiance and global illumination for adding an object in a real scene coherently. The scene is splitted into three parts: local, global, and synthetic. A light source model is created for the distant scene, and the material of the local scene is estimated. Finally, compositing using a light probe is presented.

Another application for augmented reality is proposed by See *et al.* [SBC15]. They use a camera system to capture a set of images and recreate an HDR environment map oriented to an augmented reality application. Indeed, an environment recognition for a mobile device is then used to add information on buildings present in the environment. More recently, Monroy *et al.* [MHS18] solve for an augmented reality application using a mobile device in real-time. Using a set of images and depth information, an environment map is captured, and virtual objects are rendered. Over time, the environment map is updated when the mobile device captures new data.

Grosh *et al.* [GEM07] propose an illumination method of virtual objects in a real indoor environment (a Cornell Box) illuminated by outside lighting. To obtain a coherent illumination of the virtual object, the near (considered as indirect and diffuse light) and far illumination (considered as direct illumination) are computed separately. The far illu-

mination is obtained using a fish-eye lens mounted on a camera. A coherent illumination is computed without depending on the position of the fish-eye lens compared to the object position. For this, irradiance volumes are computed for several possible emitters to approximate indirect light. The superposition principle of light is then used to obtain the final illumination.

2.5.5 Deep learning

The recent rise of deep learning, made possible by new computing facilities, has opened research to their use on environment mapping. In most of the cases a Convolutional neural network (CNN) is trained [HGSH+17] [RZZY17]. Because this research topic is recent, the training datasets used are most of the time created by the authors [ZL17] [MYM+17] [GSY+17] [WPL18] [CLG+18] [GRR+18], and can contain real and synthetic examples.

Several approches aim at estimating illumination from a single image. Hold-Geoffroy *et al.* [HGSH+17] recover a high dynamic range (HDR) outdoor illumination using a single low dynamic range (LDR) image. The CNN is first trained on a large database of 360° panoramas. It is used to extract output sky parameters of the input image. Results demonstrate good prediction of an HDR outdoor illumination which corresponds to the input image. Similarly, Gardner *et al.* [GSY+17] propose a method to recover an HDR environment map from a single indoor (low field-of-view) image. The CNN is trained on two datasets: one to train the network on the light directions, and another one to train on light intensities. The authors have created the second dataset (of 1750 HDR environment maps). In a second time, a single image is given as input, and the CNN predicts the environment map which corresponds best to the illumination of the input image. In both papers, authors demonstrate their results by showing rendered objects using the predicted environment map in the input image.

Georgoulis *et al.* [GRR+16] recover natural illumination from a single image. They expose a network architecture using sub-networks to convolve the background image and obtain a partial reflectance map. They finally de-convolve the results to obtain an illumination map. The dataset contains real and synthetic examples. Georgoulis *et al.* [GRR+18] propose another method to obtain reflectance and illumination from an object. They first obtain the reflectance map and then decompose it into two layers: the BRDF parameters and an environment map.

Figure 2.20 – Results of the horizon alignment of a panorama on the right-hand side from the input image on the left-hand side. Courtesy of [Uyt17].

Mandl *et al.* [MYM+17] use geometry and albedo of objects to estimate incident lighting in real-time. Several CNNs are trained with different illuminations and camera poses of a 3D object (one for each pose and illumination). For a given input image, the camera pose is detected first before choosing which CNN must be used to detect the correct illumination. Some results of augmented reality are finally exposed using the detected lighting and compared to the real environment map. Weber *et al.* [WPL18]) propose to predict the indoor lighting using a picture of a known object (geometry and reflectance). This method does not use a simple CNN but an auto-encoder, which is an unsupervised deep learning algorithm. The auto-encoder presented here is able to predict plausible lighting. To train this network, authors use a dataset of 21,000 HDR indoor environment maps.

Calian *et al.* [CLG+18] estimate an HDR environment map from a single LDR outdoor human face image. The idea is to use the face as an environment map. They recover 3D geometry of faces and train an auto-encoder to model a sky hemisphere. Furthermore, they present a dataset of synthetic and real faces. Zhang *et al.* [ZL17] recover an HDR environment map from an LDR outdoor panorama. They also use an auto-encoder trained on a dataset of LDR panoramas.

Uyttendaele [Uyt17] proposed a method using deep learning for aligning panoramas' horizon. Indeed, a common limitation of spherical images is that, when a user takes a picture using a 360° camera, the horizon can be deformed, which breaks realism. This article correct the deformation of such picture affected by the rotation of a 360° camera ("*tilt and roll*"). The third rotation of the camera does not distort the horizon but only changes the initial viewing (*i.e.*, the center of the photo). A DNN (deep neural network) is trained on 500,000 rotated panorama labeled with tilt and roll values to learn how to correct them. To obtain better results, several rounds of training are executed to finally obtain a rotation to within 0.1 degrees on average (result on 2.20).

Another approach [RZZY17] allows a robot navigation in an outdoor environment as illustrated in figure 2.4 top right. A Convolutional Neural Network is used to train a series of classification tasks for the navigation. The CNN is trained using 360° fish-eye panoramas.

2.5.6 Virtual visit

Spherical images have been used increasingly in the last decades to allow a user to virtually visit a place. This method provides better immersion or comprehension of the place than a set of low angle pictures. The immersion feeling can be increased with the use of immersive virtual reality technology. In most of the cases, the spherical images are taken at different places and allow the user to move from one panorama to another interactively. This kind of visits are suitable to expose to the public cities (Venezia [Gabb], Bordeaux [Ven]), famous places (Parc Güell [Gue], Angkor-wat [Str]), museums (several examples on [MdP]), interesting places (Mars [Gaba]), or historical monuments (Chenonceau castle [Ecl], Vatican [Vat]).

Other fields of activity have developed tools to allow virtual visits as the real estate domain or automotive domain. They permit interactive visits of houses or cars for the clients. We can also mention (Google street view), a virtual navigation tool launched in 2007 that allows visualising many public places all over the planet.

Koehl *et al.* [KSF+13], show the set up of a virtual tour of the municipal baths of Strasbourg and the different stages of the design, acquisition and implementation. The baths are described as *"particular buildings, their visit, their interpreting, and their access is not anymore very well-to-do"*. It is possible to have a guided virtual visit, from several view points on panoramic images from inside and outside.

Zhang *et al.* [ZZM17] propose an acquisition system for museum virtual tour. Using a 360° camera mounted on a 120cm rail they are able to take a picture each centimeter. The rail can be moved during the capture to acquire a whole room or museum. The small distance between two panoramas avoids large seams in the final stitching result. Indeed, seams appear when two images with a large offset are stitched; especially with near objects (and so object we are looking for). This device allows users to move in the environment smoothly, without uncomfortable artifacts, and look all around during the visit. Some applications using a computer screen or a virtual reality head-mounted display are shown.

2.5.7 Billions pixels panoramas

Panoramas of billions of pixels have been created to obtain high-quality images. This quality of image allows the user to zoom in while keeping a good image quality and explore the scene in detail. This kind of panorama is made up of thousands of images stitched together (see section 2.1.2.1) to compose a single image.
At the moment, the biggest image using this kind of method is an 846 billion pixels panorama of Kuala Lumpur (Malaysia) published in 2015 by the Limkokwing University of Creative Technology, using 31000 images. The same year, the in2white company [FAG+] published a panorama of the *Mont Blanc* using 70000 images to obtain a single image of 365 billion pixels (more details about the method and panorama are available on the company website).

More recently (November 2018), a panorama of 405 billion pixels of the *Old Town Square* (Prague, Czech Republic), using 8000 images have been published [Jef]. The number of images used is significantly smaller than the previous one, due to the advances made in camera quality.

2.5.8 Conclusion

In these sections, we have shown that environment maps are used in several domains, as for their simplicity of acquisition, manipulation, or representation. Their origin lays in the rendering area of graphics, specifically in image-based lighting. We showed that many research directions relied on environment maps to compute lighting, and for virtual-real coherent lighting in augmented reality. However, their use goes beyond lighting techniques. For instance, they are used for automatic navigation or remote visualization, and many other applications. Recent directions have proposed approaches to solve environmen- map-based problems with deep learning.

2.6 Environment map - Positioning

2.6.1 Introduction

These last years several methods have been developed to compute rotation of panorama. While it is now fairly easy to capture a 360 degree image, its positioning in a 3D space still relies on manual input. A few approaches have been proposed to orient automatically environment maps. It is useful for vehicles positioning or panorama registration (see section 2.6.2 or section 2.6.3).

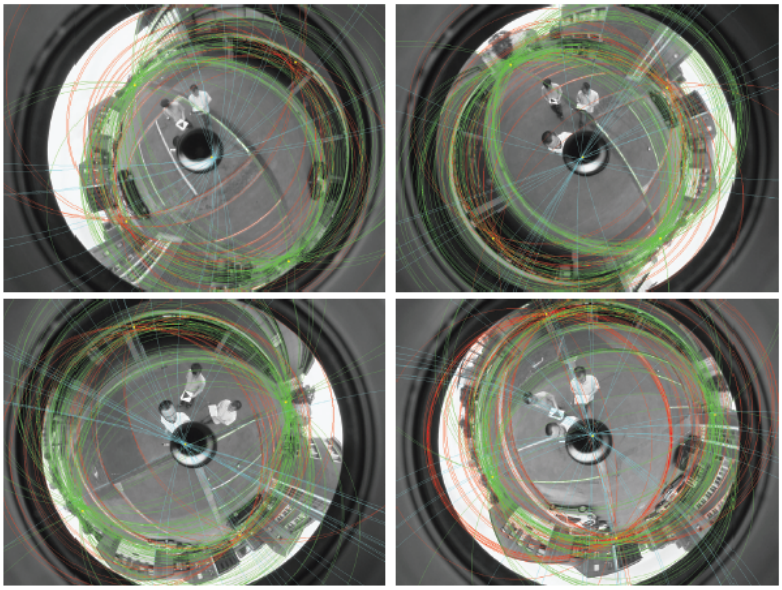


Figure 2.21 – Result of parallel lines and vanishing point extraction. Courtesy of [BDVK12].

2.6.2 Vehicle orientation

The most common usage of such task is the navigation of a vehicle in an unknown environment ([RZZY17] or [BDVK12]): the aim is to automatically orient the vehicle equipped with a 360° camera, taking a video or pictures at a regular time interval. In this configuration, orienting the vehicle is equivalent to orient the spherical image. In [SCMS08], the authors first extract vertical lines from omnidirectional images. Then they calculate a descriptor for each line in a given image. To orient the robot, they finally find matches between lines using the descriptor in consecutive frames of a video taken by the omnidirectional camera of the robot. Theses matches allow the robot to orient itself.

In [BKDV08] and later in [BDVK12], lines are also extracted to orient a robot in an unknown urban environment. In these papers, parallel lines are extracted in different directions and related to vanishing points, as shown in 2.21, using a rotation estimation (top-down approach). They calculate a rotation matrix to recover the complete rotation of the omnidirectional images of a video sequence in real-time.

In [RZZY17], robot navigation, as seen in figure 2.22, uses a Convolutional Neural Network to train a series of classification tasks for the navigation. The CNN is trained with 360° fish-eye panoramas. Makadia *et al.* [MSD04] [MD06] solve for vehicles navigation using omnidirectional images. The used the spectral domain and compute Fourier

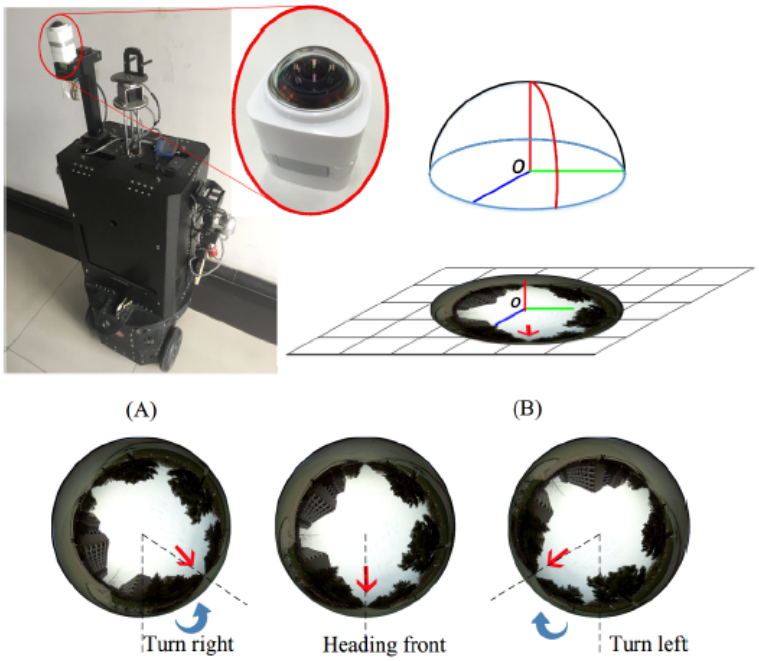


Figure 2.22 – A robot vehicle (A), the spherical camera (B) and spherical images. Courtesy of [RZZY17].

coefficients in two images to recover the rotation of the images.

2.6.3 Panorama registration

In [DE02],a method is described to register an image with a small field of view and a high resolution with a panorama at low resolution, as shown in figure 2.23. They compute a set of simple features (gradient) that are compared in both images. Then they compute the homography which minimizes a correspondence error calculated from the features.

Figure 2.23 – Registration results of the small field-of-view image with a low resolution panorama. Courtesy of [DE02].

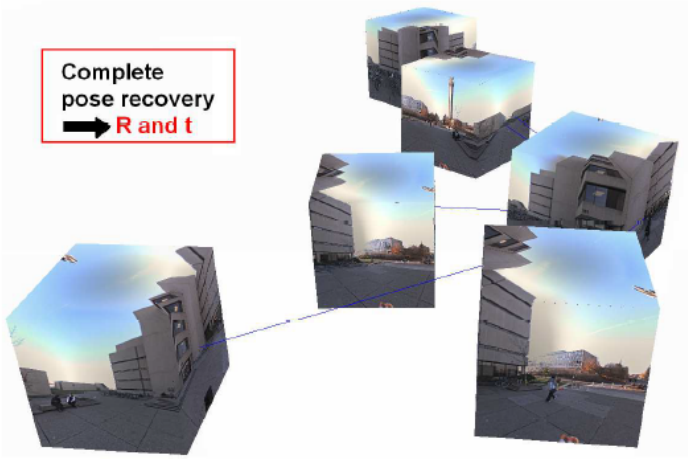


Figure 2.24 – Pose recovery. Courtesy of [LK10].

Another method, proposed by Laganière *et al.* [LK10] orients and estimates the position of a dataset of panoramas in an environment. The first stage of the method uses epipolar constraints to find the panoramas alignment for each panorama. The second stage of the method proposes to recover the 3D position of each panorama relatively to others (figure 2.24). They show results for indoor and outdoor scenes.

Benseddik *et al.* ([BHACB16]) show a method using 3D mesh generation of spherical image to estimate spherical image rotation. Authors first map the panorama on a sphere and generate a 3D mesh using shape analysis (SPHARM tool). After fixing potential errors by filling holes that may be present in the mesh, they compute the optimal orientation.

2.6.4 Conclusion

In this section, several method to register an environment map are shown. This type of methods result to the necessity to know the orientation of an environment map to use it to obtain correct result. We saw this necessity as for vehicle orientation in an unknown environment than for the recovery of a 3D model of a scene. Indeed, to correctly drive, the vehicule must know the precise path to take, or to reconstruct the environment model, we must be able to replace precisely the different object present in the scene.

2.7 Conclusion

In this chapter, we saw the different environment map aquisition ways, there use in the rendering domain, their usage and different methods where environment map orientation

is important.

First the different ways to obtain an environment map have been shown. Two classes of methods are discernible. The first one is the creation of an environment map using a set of low field-of-view images and the use of a stitching method to obtain the high field-of-view image. These methods present an higher computation cost, and stitching and holes problems can appearss but allows the user to controle the panorama creation parameters (as the angle of view, change a part of the final image if an undesirable artifact appear). The second class of methods is the use of aquisition devices which provide an all-in-hand way to obtain such image. Even if a stitching step is done in some devices, the user does not have the hands on it and directly obtain a large field-of-view image.

Secondly we show the different kind of environment map representation in computer to estimate which representation is privileged in the diferent domains. We show that some representations (*e.g.*, cube map) are preferable to real time application because presenting a better access time. Other representation are preferable to user understanding because less distorted (*e.g.*, equirectangular map). A comparision of texture representation is presented in table 2.1.

Finally, we show different environment map usages to exhibit the compunity lure and present the different representation interest. More, we show that environment map orientation and positioning are important in different area (as 3D reconstruction or orientation in environment) to obtain correct results.

Chapter 3

Framework for automatic environment map orientation around a local scene

Résumé

Dans ce chapitre, nous proposons une méthode d'orientation automatique de cartes d'environnement (*environment map* ou *EM* en Anglais). Pour obtenir un éclairage de scène cohérent, aussi bien pour le ré-éclairage ou la représentation de matériau comme dans [BCRA11], il est nécessaire d'orienter la scène globale représentée par une carte d'environnement, avec la scène locale. Ainsi, nous présentons dans cette partie un préambule à notre méthode d'orientation automatique de cartes d'environnement avec une connaissance minimale de la scène locale.

Le but de cette approche est de trouver l'angle d'orientation (theta, dans la figure 3.1) de la carte d'environnement afin d'obtenir un profil lumineux cohérent par rapport à la scène locale. Dans les faits, nous retrouvons aussi le rayon de la carte d'environnement et donc une position correcte de la source lumineuse principale, mais cette valeur n'a que peu ou aucun sens physique. Notre volonté est de proposer une méthode utilisable avec toute carte d'environnement et toute scène locale. Notre but est d'utiliser le moins d'hypothèses possibles sur les données d'entrée.

Une première approche utilise l'information lumineuse de la scène et trace des rayons qui suivent le trajet lumineux dans la direction des ombres. La projection de rayons place au sol de la scène locale, selon les différentes méthodes présentées, une zone d'ombre au sol ou une zone de lumière sur la carte d'environnement. L'étape suivante vise à faire correspondre, respectivement, l'ombre ou la lumière identifiés dans les données d'entrées avec celle calculée.

Nous présentons d'abord les données d'entrée de notre méthode dans la section 3.2 : nous présentons la scène globale 3.2.1, la scène locale 3.2.3 et les informations nécessaires qui leur sont associées (la position de la source lumineuse principale et ses masques). Ensuite, nous présentons dans la section 3.2.5, les informations de profondeur et les informations géométriques d'un objet, qui sont deux données d'entrée mutuellement exclusives qui concernent un objet de référence présent dans la scène locale.
Ensuite, nous présentons dans 3.3, l'idée générale de la méthode et les principales étapes de celle-ci : la représentation 3D de la scène qui nous intéresse, les étapes de traitement et de minimisation et le résultat final.
Enfin, nous présentons les différentes définitions de rayons possibles (voir section 3.3, différent *light path*) utilisées par notre méthode. En effet, parce que notre méthode utilise le

chemin de la lumière, nous pouvons définir différentes façons de suivre celui-ci, chacune de ces définitions étant liée à la façon de créer la représentation 3D de la scène et d'utiliser les données d'entrée. Les différentes définitions des rayons peuvent utiliser des données d'entrée différentes, mais la plupart du temps, les données utilisées sont identiques. Si des données d'entrée supplémentaires sont utilisées pendant l'étape de définition du rayon, nous les explicitons.

3.1 Introduction / Context / Problematic

In this chapter, we present a framework to orient environment maps (*EM*) automatically. When used to obtain a coherent scene illumination, for example in relighting or material depiction [BCRA11], it is necessary to orient the global scene, represented by an environment map, with the local scene, represented by a low field-of-view image taken from an arbitrary viewpoint. We present in this part a preamble to our automatic environment map registration method.

The goal is to find an orientation angle (theta, in figure 3.1) of the environment map to recover a correct light profile regarding the local scene. In fact, we also recover the environment map radius to obtain a correct main light source position, but this value does not have any physical meaning. Our will is to propose a method, fully automatic, usable with most environment maps and any local scenes. Our goal is thus to use the less possible assumptions on input data and not to restrict our input data with specificity like resolution restriction. Our method uses the light information of the scene and traces rays along the light path to recover the orientation using shadows. The ray projection defines a shadow area on the ground or a light area on the environment map. The next step aims to match the input shadow with the computed shadow or the input light position with the computed light position.

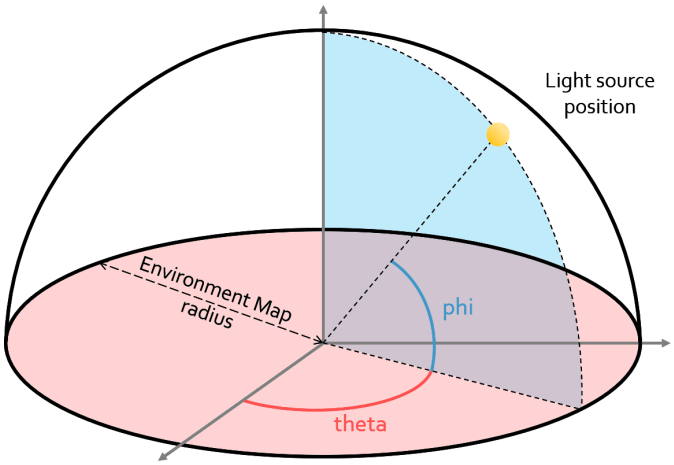


Figure 3.1 – Environment map scheme. Theta and phi angles.

Firstly, we present the input data in section 3.2 the different parts of the scene: we present the environment map also called global scene (section 3.2.1), the local scene represented by an object (section 3.2.3), and the necessary information associated such as the main light source position (reduced to a point) and local scene object mask. Then, we

present in section 3.2.5, the depth information and the geometry information of an object, that are two mutual exclusive input data which concern a reference object of the local scene.
Secondly, we present in section 3.3, the global idea of the method and the main steps: the 3D representation of the scene we are looking for, the processing and minimisation steps, followed by the results we obtain.
Finally, we present the different possible ray definitions (in section 3.3) used by our method. Indeed, each light path is linked to the manner the 3D representation of the scene is created and the input data is used.

3.2 Input data

Our input data are a local scene image, a global scene image, and some knowledge of the geometry of an object, called the reference object in the rest of the document present in the local scene image taken from an arbitrary viewpoint. The global scene is represented by an environment map, ideally located at the center of the local scene. In the environment map, we should also see all the light source components (connected area). Moreover, we assume that we know the mask of the shadow of the object present in the local scene and the position of at least one light source (which we reduce to a point) in the environment map. These assumptions are reasonable because there exist many methods in the literature to extract these information.

3.2.1 Global scene : Environment map

The first input data of our method is a 360° degree image of the scene. As seen in the state of the art (see chapter 2), there are many ways to obtain an environment map. All kinds of environment maps are possible as input data, and their representation can vary (sphere, half-sphere, cylinder, or cube map, for example). Because we use the environment map to recreate a 3D representation of the scene and we project the environment map image on an half-sphere, any representation is possible. Only an adaptation to obtain the correct projection is needed.
The only necessary information we need to know about the environment map is the position of a light source, either by its coordinates in the image or by its area. In this work, we use the *equirectangular* representation, in which the theta angle directly corresponds to a longitude coordinate on the horizontal axis.

Figure 3.2 – Global scene or environment map (top) containing a visible light source or main light source. Mask of the main light source (bottom).

3.2.2 Light source position in the environment map

Our method uses the light paths to guide the orientation of the global scene around the local scene. For this, it is necessary to know the position of at least one light source in the global scene (see figure 3.2) corresponding to a shadow of the local scene. It is most of the time, the main light source of the scene.There exist several approaches to estimate light source positions from images (see **??**).
However, it is not always possible to obtain the light source position in the environment map image (if hidden or too diffuse). In this case, the center of the brighter part of the image would be considered as a light source.

3.2.3 Local scene

A small field of view image defines the local scene (see figure 3.3, left). This image must contain an object (called the reference object) from which we know two kinds of information: its shadow mask and its geometry. In the method explained in 4, the main idea is to use the 360° camera as the reference object of the local scene we use. That presents

Figure 3.3 – Local scene containing the reference object and the shadow corresponding to the light source of the environment map (left). Mask of the shadow and of the reference object (right).

several advantages:

- As described in section 5.5.3, we need to know the exact 3D of the object we consider. Using the camera and not a different object each time allows us to define only one 3D model and not to have to recover it.
- We do not have to choose a reference object in the scene or to remove it (or move it) to take the 360 picture. Moreover, if the object is not moveable the center of the environment map will not be exactly at the same place as the reference object. That will generate a gap between the two centers (center of the environment map and center of the object) and introduce an error during the orientation.
- Another advantage is that we can obtain the data easily by placing the 360° camera and then using a single device to obtain in a few seconds the image (local scene) and the environment map image (global scene). This way, we ensure that the illumination of the scene will not change during the capture.

We assume that around the reference object, the ground is flat and horizontal. More precisely, the shadow of the reference object must be projected on flat ground to obtain correct results during the orientation step. Since the shadow of the reference object is small (if a 360° camera is used, it has a measure around ten centimeters), the assumption is reasonable.

3.2.4 Reference object and shadow masks

Another important input data is a mask of an object called *reference object*, present in the scene, and a mask of its shadow (see figure 3.3, right). It is necessary to recover the two

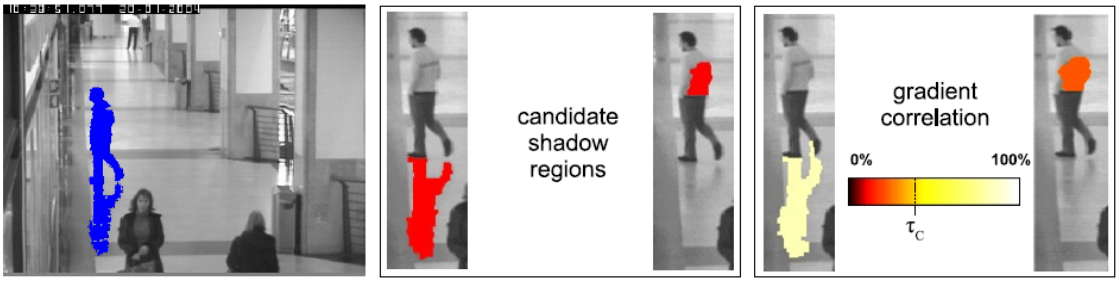


Figure 3.4 – Result of the shadow detection (right image, area in white) using gradient method. Courtesy of [SSL10].

masks from the local scene image which can be done using image processing approaches. As for the light source position in the environment map, there are many methods that allow us to obtain the shadow easily. In order to detect shadows, several approaches were proposed. A survey [SSL12] classifies methods according to four criteria: methods based on scene geometry, chromaticity, the physical property of objects of interest (*e.g.*, the kind of object for which we want to obtain the shadow), or the object texture. Each group of methods is compared along the following criteria: execution time, the possibility of using various objects, the importance of the assumptions.

A second survey [ANBSE12] exposes different types of methods according to the dependency to the object or scene interest, of the level of precision (pixel or image-level), or according to the usage of the color or not. Each group of methods are compared against their dependency to the scene, the accuracy of their results, the consistency of the detected area, and their execution time.

In [SSL10] the direction of the gradient is used to detect shadows in surveillance videos. Lalonde *et al.* [LEN10] assumes that the materials constituting the ground and thus the shadows are limited. They first proceed to a learning phase on a dataset and finally test a real situation. Wu *et al.* [WT05] use a Bayesian approach to define if each pixel of the image is an umbra pixel or not.

3.2.5 Geometry information

A last input to our method is the 3D information of an object. We call it the reference object. Needful information about the local scene is the 3D geometry of an object and its shadow (see figure 3.3, right). Using depth sensors, a partial geometry of the object can be recovered. We describe in section 4.2.1.1 how we recover it and describe advantage and

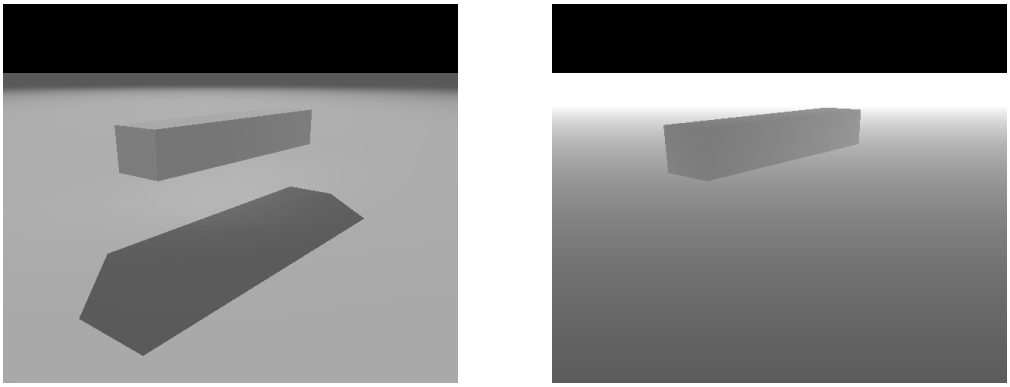


Figure 3.5 – Depth information (right) of a local scene (left).

inconvenient of such reconstruction.

It is possible to use the 360° camera itself as a reference object. Several examples using the 360° camera as the reference object are shown in chapter 4. Because the two images of the global and local scene are taken from the same scene, and so at the same moment, but with two different devices, the reference object in the local scene image can be the 360° camera which is already present in the scene. We can notice the, in this case, the reference object is not present in the environment map and can not be recovered in the global scene image. We have created a 3D model of our 360° camera (see figure 3.6, right). The goal of this representation is to represent the global shape of the 360° camera and not all the details. A simple model is sufficient. The model has only 303 vertices, 605 edges, and 306 faces. If another device is used for the capture of a global scene or if the chosen reference object presents another 3D shape than the one we present in this manuscript, we can easily replace the 3D model to adapt our method.

It presents several advantages:

- By definition, the 360° camera is already present in the scene. Indeed, if we capture the global scene and the local scene at the same time, we first have to place the 360° camera in the scene to capture the environment. Then, we capture the local scene. Thus, it is better not to remove the 360° camera from the scene but to focus on it.

- In this way, we already knew the geometry of the reference object, and we do not have to estimate it using estimation methods or depth methods.

- Most of the 360° cameras have a simple geometry easily reproducible with a 3D software. Our test shows that even an approximation by a simple geometry form like a rectangular parallelepiped (shape of the used camera) is sufficient to obtain correct

Figure 3.6 – Picture of the 360° camera. A simple 3D model of the reference object we use (created with blender): 303 vertices, 605 edges, 306 faces).

results.

- The reference object does not change from one scene to another. Indeed, if we do not use the same reference object, we need to create a 3D model database and select a correct one each time.

However, it is possible to use another reference object than the proposed one. This reference object need, however, to present a single characteristic: the reference object should be not too small. The reference object should present an elevation compared to the ground to obtain a correct reference shadow. Indeed, our method uses shadow shapes comparison to register the environment map. Thus, it needs to be large enough to be significantly discriminant for our computations.

3.2.6 Input data: generalities

In the different methods presented here, the presented input data are not all used, especially for the methods trying to reduce the number of input data (see chapter 5). However, there are not huge input data changes from a method to another. We explicit each time it is necessary for each method the input data used.

We have chosen input data with the minimum assumptions on it. We have tried to reduce the input data and a small diversity of the types of data. Thus, we do not want to have an assumption on the camera parameters, *e.g.*, its position relatively to the reference object or the focal length. This is because such an assumption could easily be not fulfilled. Indeed, in some cases, we could have some situations where the chosen reference object

could be placed at an arbitrary place (*e.g.*, a scene static object like a statue) or could not be approached by the user to be photographed.

3.3 Method overview

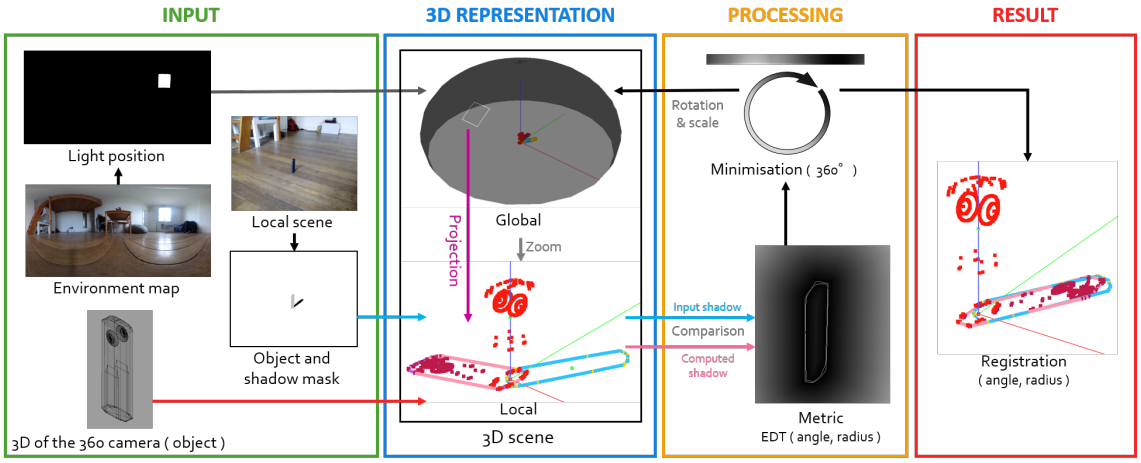


Figure 3.7 – Overview of the different steps of the presented method.

Input data: We present here the global idea of the method. We first use the input data (green rectangle in figure 3.7) to obtain the main light position in the environment map and the reference object masks. On the figure above, not all the input data are presented. For example, the depth information is not represented in this figure. But the 3D of the camera represents it, because used the same way. In other words, because the depth information of the scene and the reference object geometry give us the same information, only one of them is present here.

3D representation: Using the input data (we can not specify them here as described in 3.2.6), we create a 3D representation of the scene (blue rectangle in figure 3.7). This representation includes the global and the local scene, including the light sources and surrounding objects present in both local and global scene. The global scene is represented by a half-sphere (top of the blue rectangle of figure 3.7) using the environment map image. The local scene is represented in the center of the global scene sphere (bottom of the blue rectangle of figure 3.7) using the local scene image and local scene masks to obtain the input shadow (blue shape in the local representation of figure 3.7) and the reference object geometry to directly obtain a 3D representation of it (red points in the local representation of figure 3.7).

To complete the 3D representation of the scene, we use the main light source position present in the environment map to project the shadow of the reference object on the ground (purple shape in the local representation of figure 3.7) and obtain a computed shadow shape.

Processing, minimisation & results: Using these two shadow shapes (the input one and the computed one), we proceed a minimisation step of an Euclidian Distance Transform (EDT) metric (yellow rectangle in figure 3.7). This EDT metric (on the bottom) estimates the difference between the two shapes. We test different positions for the main light source (*i.e.*, different environment map rotations) to find a match between them (position where the metric is low) and recover the correct orientation.
We estimate that we have recovered the correct position (red rectangle in figure 3.7) of the environment map when the two shapes coincide and when we found the minimum value for the EDT, thus indicating that the computed shadow matches with the input shadow.

We define a ray as a *segment* between two points (*i.e.*, pixels in the respective images) belonging to two different entities among the main light source position in the environment map, the reference object, and the reference object shadow. So, we do not consider multiple bounces for a ray but only a single segment.
Because we aim to compare shadow shapes, we only consider the boundaries of the three entities (object silhouette, shadow boundaries and light source outline). We trace several rays, *i.e.*, a *set of rays* considering the boundaries, thus gaining in efficiency. A second advantage is that we obtain fewer rays and so, less calculus has to be done to get a result. We compared three different strategies, solving for light, shadow or object position. Thus, we defined rays along three different paths:

Path 1: from the shadow to the object

A first way to define a ray is to consider a segment between a point of the reference object shadow and of the reference object. Thus, such a ray is defined totally in the local scene. It is projected into the environment to link the local scene to the global scene. The method to choose which point of the shadow and the object match is defined in section 5.2.1.
The projection of a static set of rays on the environment map surface (*i.e.*, the environment map intersection with each ray) define a coherent shape, the *silhouette* of the light source area. The goal is then to rotate the environment map to match the main light source position with the obtained shape.

Path 2: from the shadow to the light

A second way to define a ray is to consider a segment between the main light source in the environment map and a point of the reference object shadow. The ray directly links the global scene to the local scene.
This defines a set of dynamic rays, which position varies with the position of the main light source, and so, changes if we rotate the environment map around the local scene.
When rotating the environment map, it leads the set of rays to coincide with the reference object position in the local scene. Because we only consider the boundaries of the reference object shadow, we obtain an empty cone determined by the rays inside which the reference object boundaries need to lay.In this case, we can notice that we do not directly define rays between a point of the object and a point of the shadow, which is an essential step of the previous ray definition method. No projection has to be done (as in the environment map as in the path 1, or in the ground as in the path 3) and the object position is necessarily placed on the cone thus defines.

Path 3: from the light to the object

A third way to define a ray is to follow the light path. We consider the segment from the light source in the environment map to the reference object in the local scene. A shadow shape is obtained by projecting the rays on the ground. We obtain a set of 2D points. Their convex hull defines the shadow boundaries. Used this way, the set of rays is not static. Indeed, it depends on the position of the main light source, and so, changes if we rotate the environment map around the local scene.
The goal is then to rotate the environment map to modify the dynamic set of rays in order to make the projected shadow silhouette coincide with the original shadow shape position in the local scene. In this case, we finally compare two 2D objects: an original (or input) shadow shape and a computed one. The correct position of the environment map is defined by the optimal overlap of both shapes.

3.4 Discussion

In this thesis, we propose several methods to solve environment map registration with a local scene. Each starts with one of the three definitions of the ray paths as defined in this chapter. These methods are explained in the two next chapters. The table 6.1, section *Discussion* (5.5) show a summary of which ray definition is used for the different tested methods.

Chapter 4

A full 3D method for automatic environment map orientation

Résumé

Introduction : Nous présentons ici l'approche finale de notre méthode d'orientation. Cette méthode utilise une représentation 3D de l'objet de référence présent dans la scène locale et le chemin de lumière allant de la source lumineuse présente dans la carte d'environnement vers l'objet de référence (voir section 3.3, light path 3), pour projeter au sol une ombre calculée, et comparer itérativement avec l'ombre de référence présente dans la scène locale.

Nous avons développé deux méthodes utilisant des données d'entrée différentes. La première utilise l'information de profondeur de la scène locale pour recréer une géométrie partielle de l'objet de référence et l'utilise alors pour calculer l'ombre. La seconde utilise directement en entrée un modèle 3D simplifié de l'objet de référence pour obtenir la géométrie de l'objet et calculer l'ombre.

En utilisant l'ombre calculée, nous n'avons plus qu'à comparer des formes 2D, celle d'entrée présente dans l'image de la scène locale et celle que nous venons de calculer. Si les deux formes coïncident, nous pouvons considérer que la source lumineuse présente dans la carte d'environnement est à un endroit correct, et alors admettre que la carte d'environnement est bien orientée.

Information géométrique de l'objet de référence : La première étape de cette méthode est de récupérer l'information géométrique de l'objet de référence que nous utilisons. Pour cela, nous présentons deux manières d'obtenir l'information 3D. La première manière est de capturer une image de profondeur de la scène locale, qui correspond à l'image d'entrée (voir section 3.2.3). En s'aidant du masque de l'objet de référence et de la carte de profondeur, nous sommes en mesure d'extraire l'information de profondeur de l'objet de référence. A l'aide d'une méthode similaire à celle décrite dans la section 5.2.1.1, nous obtenons un ensemble de points 3D pour l'objet de référence.
La seconde manière est de créer un modèle 3D simple de l'objet de référence et de le fournir en tant que donnée d'entrée de la méthode. La seconde manière est plus efficace. En effet, la méthode déduisant les coordonnées 3D à partir d'images est d'une part moins précise, et d'autre part ne permet que la récupération des points de l'objet visible par la caméra ayant capturé la scène locale.

Définition des rayons : Une fois l'ensemble de points 3D de l'objet de référence obtenu, nous plaçons celui-ci au centre de la représentation 3D de notre scène. La définition des

rayons suit alors le parcours de la lumière : nous traçons des rayons de la source lumineuse présente sur la carte d'environnement jusqu'à chaque point 3D de l'objet de référence. Nous projetons finalement ces rayons sur le sol (plan $z = 0$) pour obtenir un ensemble de points 2D (de même taille que le nombre de points 3D de l'objet de référence).
Pour plus de clarté et pour réduire le nombre de calculs fais, nous calculons l'enveloppe convexe de l'ensemble de points 2D.

Orientation de la carte d'environnement : L'ensemble de points 2D alors obtenu est alors considéré comme *l'ombre calculée* de l'objet de référence. Nous pouvons comparer cette ombre avec l'ombre d'entrée fournie par l'image de la scène locale. Nous pouvons considérer que si l'ombre calculée est la même que l'ombre d'entrée, alors la source lumineuse est à la bonne position, et donc que la carte d'environnement est correctement orientée. Puisque les rayons sont définis en fonction de la position de la source lumineuse, leur projection et l'ombre calculée l'est aussi. En faisant varier l'orientation de la carte d'environnement, nous pouvons donc comparer plusieurs ombres calculées avec celles d'entrée.

Résultats : Pour définir la position correcte de la carte d'environnement (*i.e.*, quand les ombres coïncident), nous calculons une métrique. Cette métrique calcule la différence entre deux formes d'ombre. Cette métrique repose sur une carte de distance (*Euclidean Distance Transform* ou EDT).
Nous montrons que nous obtenons des résultats cohérents pour orienter une carte d'environnement autour d'une scène locale et présentons finalement des résultats afin de valider notre approche.

Conclusion, limitations et travaux futurs : La dernière partie de ce chapitre est consacrée aux limitations de notre approche et aux différentes améliorations possibles que nous pouvons lui apporter pour améliorer les résultats tant en précision, qu'en temps de calcul.
Nous concluons finalement la démarche générale de notre approches développée dans les chapitres 3, 5 et 4.

4.1 Introduction

We present in this chapter a first automatic approach of our orientation (*i.e.*, registration) method. This method uses a 3D representation of the reference object present in the local scene, and the light path from the light source present in the environment map to the reference object (*i.e.*, section 3.3, light path 3), to project on the ground plane ($z = 0$) a computed shadow, and compare it iteratively with the reference one present in the local scene.
We have developed a method able to use different input data. One using depth information to recreate a partial reference object geometry and using it to compute the shadow. The second one uses directly as input a simple 3D model of the reference object to obtain the geometry information and calculate the shadow.
Using the computed shadow, we must only compare 2D shapes, the input one present in the local scene image and the computed one. If the two shapes coincide, we can consider that the light source present in the environment map is at a correct place, and so, consider that the environment map is well oriented.

The algorithm (algorithm 28) presents the main steps of our method. It shows step by step the different parts of the method and explicit which input data are used and in which dimension (2D or 3D) the computation is done.
The input data (line *Data*) are the environment map image (EM), the light position point in the environment map (p_light_pos_{2D}), the image of the shadow mask (SDW_{mask}),the image of the object mask (OBJ_{mask}), and the 3D points of the reference object ($3D\_object$).
The first loop (line 3) computes the 3D input shadow points. The three next lines (7, 8, 9) extract the convex hull from this set of points to obtain the reference shadow. An image of this convex hull and a Euclidean Distance Transform (EDT) image from the convex hull (*i.e.*, im_EDT_input) are then created. It is more relevant to compute the convex hull of the set of points because only the shapes will be compared. The points inside the hull will bring more calculus but no more precision.
The two next loops (lines 11 to 25) are the minimisation steps of the method. The goal of this step is to find the best *theta* angle (*i.e.*, $best\_angle$ in the algorithm) for the environment map. The first for loop (line 11) modifies the orientation of the environment map. We test each possible angle around the z-axis. The experience shows us that the testing angle under one degree is not relevant. The second for loop (line 12) modifies the scale of the environment map. We test different possible radius between two high values. We do not check small radius values of the environment map because this last one represents

Data: image EM, point$_{2D}$ p_light_pos$_{2D}$, image SDW$_{mask}$, image OBJ$_{mask}$, vector<point$_{3D}$> 3D_object
Result: best_angle

```
1  best_angle, best_radius, number score ← ∞
2
3  forall point_2D p in SDW_mask do
4  |  point_3D p_i ← 3D_recovery(p, OBJ_mask)
5  |  vector<point_3D> sdw_i ← add(sdw_i, p_i)
6  end
7  vector<point_2D> sdw_input ← Convex_hull(sdw_i_z=0)
8  image im_sdw_input ← image_of(sdw_input)
9  image im_EDT_input ← EDT(im_sdw_input)
10
11 for i=0 to 360, step 1 do
12 |  for j=200 to 500, step 10 do
13 |  |  point_3D p_light_pos_3D ← 3D_pos(p_light_pos_2D, i, j)
14 |  |  forall point_3D p in 3D_ object do
15 |  |  |  r_3D ← Ray(p_light_pos_3D, p)
16 |  |  |  point_2D p_floor ← Floor_extended(r_3D)
17 |  |  |  vector<point_2D> sdw_c ← add(sdw_c, p_floor)
18 |  |  end
19 |  |  image im_sdw_c ← image_of(Convex_hull(sdw_c))
20 |  |  image overlap_sdw ← im_EDT_input * im_sdw_c
21 |  |  number score_ij ← ∑_pixels_values overlap_sdw
22 |  |  if score_ij < score then
23 |  |  |  score ← score_ij
24 |  |  |  best_angle ← i
25 |  |  |  best_radius ← j
26 |  |  end
27 |  end
28 end
```

Algorithm 1: Computation of the best angle to register the environment map. Each necessary time, we explicit when we use *2D* or *3D* element.

the far scene and so there is no meaning to test small radius. Moreover, when the radius is small, the method becomes unstable: a little orientation variation implies huge result modification. We vary this value with a significant step to avoid testing a too large number of possibilities of couple *(orientations, scale)*. Indeed, this value has no physical meaning, and the experience shows that a small difference when the scale is large, implies little or no result difference.

For each couple *(orientations, scale)*, we repeat the same processes. First, we trace rays (line 15) from the main light source to every point of the reference object (placed at the center of the environment map). Then we project these rays on the floor (line 16) to obtain a set of shadow points. We can compare the convex hull of these computed shadow shapes (line 19) with the input one (from line 7) using the EDT metric (line 20, 21). The EDT is a metric that allows us to quantify the two shapes differences: we obtain a score. Second, we minimise this score (lines 22 to 26) to get the best similar shadow shapes and so, the best orientation.

The different cited functions refer to parts of our method described in the following sections (*3D_recovery, Convex_hull, EDT, Ray, Floor_extended*). The *3D_pos* function computes the 3D position of the light source from the known 2D position, considering the environment map rotation and radius, and the *image_of* function calculates a mask using a vector of 2D points. The * operation multiplies images pixel per pixel.

4.2 Ray definition using the geometry information

We describe in this section how rays have been defined for each method. The first step of the rays definition is to recover each 3D point of the reference object geometry representation. The second step is to place the 3D points in the center of the 3D representation of the local scene. Because the environment map is considered representing the far environment, placing the local scene object at the center of this one allows simulating this. Then, rays can be defined from the light source position on the 3D environment map surface to each recovered point. Because it is impossible to determine at this step which 3D object points define the boundaries of the object, and so of the shadow, we must consider every point of the object geometry.

4.2.1 First step

The first step of the method presented here is to obtain the 3D geometry of an object of the local scene. This additional information allows us to get more precise 3D rays. We have tested two different ways to obtain this information. The first one is a recovery of the 3D information using the input object mask, the depth information of the local scene, and simple assumptions about the geometry of the scene. The second one directly uses a 3D information given as input. The second way has been chosen in the final method for its precision, accuracy results, and simplicity.

4.2.1.1 3D recovery of local scene using depth

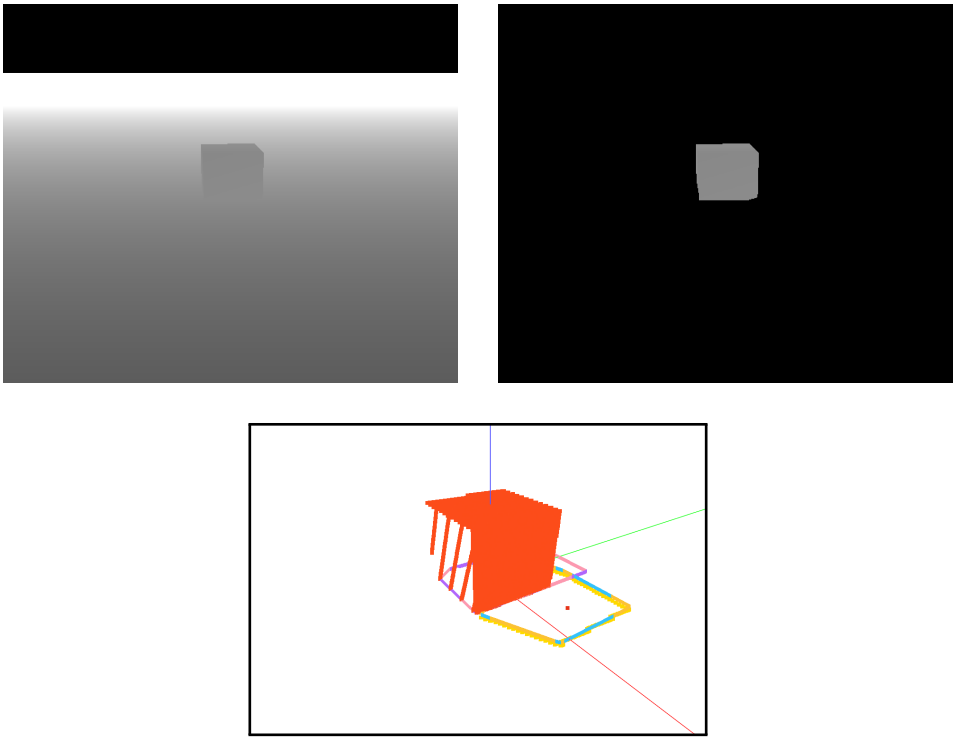


Figure 4.1 – Recover rough geometry (right) of a cube using depth map image of the local scene (left) and the mask of the object we (middle).

To obtain a rough geometry of the local scene, we use a depth map (figure 4.1, top left) and the reference object mask. This depth map has to be given as input. Despite requiring an additional device for this input, depth cameras are nowadays conventional in the consumer market and allow us to capture this information.
To obtain the geometry of the object only, and not of the entire local scene, we apply the object mask values on depth map values. We thus obtain an image with depth value at pixels representing the object and zero values for the rest of the local scene (see figure 4.1,

top right).

To recover a rough 3D information (*i.e.*, (x, y, z) values for each pixels), we consider x and z values as the value of the pixel in the image (i, j). To obtain an object center placed at the environment map center, we consider the object bounding box. We subtract the value of the center of the bounding box of the object to x and subtract the value of the lower pixel of the bounding box for z $(= z_{min})$. The z_{min} value allows obtaining an object laid on the floor and not centered around zero (*i.e.*, the lower part of the object under the ground level).

$$x = i - (max(bouning\_box_i) - min(bouning\_box_i))/2 \tag{4.1}$$

$$z = j - min(bouning\_box_j) \tag{4.2}$$

Finally, the y (*i.e.*, depth) value is equal to the depth value of the pixel minus the mean value of the depth:

$$y = depth_{pixel(i,j)} = value_{pixel(i,j)} - (max\_depth + min\_depth)/2 \tag{4.3}$$

Doing that, we obtain an object centered at $(0, 0, z_{min})$ as seen in figure 4.1, bottom: the center of the basis of the object is on $(0, 0, 0)$. The axes x, y, and z are shown respectively in green, red and blue. The object 3D points are represented in red.
We obtain a 3D point per pixel present on the object mask. We can see that the front face and top face of the cube have a dense reconstruction. However, the left face side presents only sparse lines. This is due to the position of the camera in relation to the cube. This face is represented with fewer pixels than the front and top faces. Intermediate values are not represented on this face, causing holes. Moreover, we can add that faces missing in the local scene images (and in the depth image, *e.g.*, back or right faces) are not reconstructed.

This reconstruction method allows us to work without camera parameters. Indeed, because we want a method that works with every device, we cannot be dependent on it, and more precisely, to the parameters of a unique device. Doing that exempts us from knowing or recovering the camera parameters if the device changes from capture to another.

4.2.1.2 Known 3D of the local scene

Another possibility is to give the geometry information of the local scene as input data. Because we do not use all the local scene but an object inside it, we can model it with a

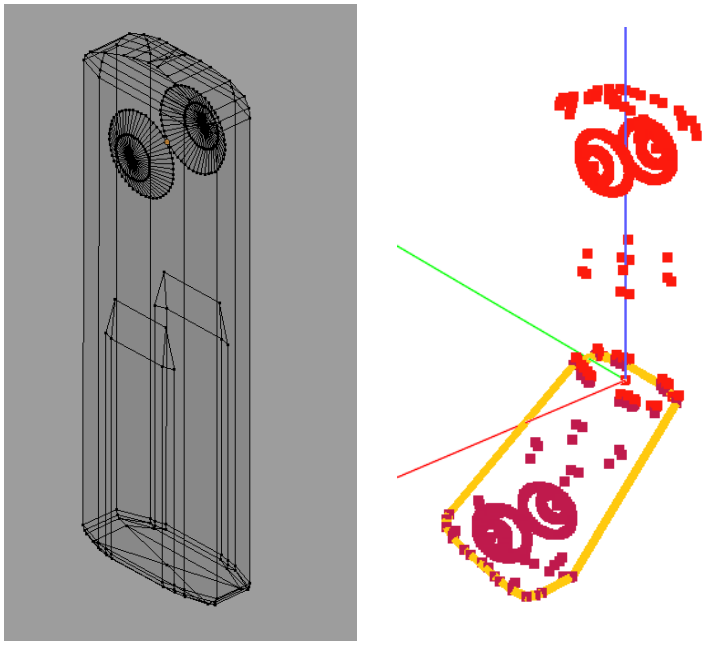


Figure 4.2 – 3D model (303 vertices) of the 360° camera (left), and 3D representation of the local scene (right): 3D points corresponding to vertices (light red) and projection of each 3D points on the floor ($z = 0$) in dark red. The orange area defines the shadow form we use in the next part of this method.

dedicated software as Blender[1] or Maya[2]. As the goal is not to obtain a perfect geometry but rough information of the shape of the object, we do not need a complex representation of the object. Moreover, because we use the 3D object points to recreate a shadow shape and compute a convex hull, we need the extreme vertices and not vertices included in an already existing face. In other words, for this method, we need vertices but not edges or faces.
Using such a model gives us directly the needed 3D points without any additional computation, only a supplemental input data (*i.e.*, the 3D model). However, this one does not change and remains the same.

We propose here to use the 360° camera itself as the 3D reference object we use to obtain environment map orientation. We use here a Ricoh Theta S camera[3]. In this perspective, we have models a 3D representation (using high-quality pictures and characteristics present in the official documentation[4]) of the 360° camera we use. This representation owns 303 vertices. Our tests have shown that we can use models with fewer vertices. But, to ensure

[1] Blender website
[2] Maya website
[3] Ricoh Theta website
[4] Ricoh Theta S Technical page

correct results in all cases, we decide to use a bigger, but more precise, model.
The advantages are that we know the dimensions of the camera we use, and we can create a correct 3D representation of the camera/object. Moreover, we do not have to choose an object in the scene or to remove it (or move it) to take the 360° picture. Another advantage is that we can easily obtain the data by setting the 360° camera in the local scene. Using a single device we obtain, in few seconds only, the local scene image and the environment map image (using the given application for smartphone for the 360° camera).
The figure 4.2 shows the 3D model of the 360° camera we use on the left and the 3D resulting points (in red) on the right side. A scene using the 360° camera is visible in figure 3.3 on page 80. At the end of this step, whether with the method using depth or the method using a 3D model, we obtain a set of 3D points placed at the center of the scene.

4.2.2 Rays definition using geometry information of the reference object

To define rays using the geometry of the local scene, we preferred to follow the path of the light (see section 3.3, light path 3). Thereby, we obtain a more physically realist method than the previously described ones by projecting a computed shadow on the ground and do not describe an area which aims to contain the light source or the object.

Initialisation: We define rays from the light source position (reduce to a point) on the environment map surface to every vertex (or 3D point) present in the 3D model. We obtain one ray per vertex. Figure 4.3 shows the rays on the local scene (turquoise color). The rays are coming from the light source position (not present in the figure) to every vertex in the 3D set (red color).

Projection on the ground: To finalise the rays definition, we project the rays on the ground to obtain a set of 2D points. For each ray, that we can describe as a line defined by two points, the light and the object point, we compute its intersection with the plane $z = 0$. In figure 4.3, the rays are already projected on the ground. The points resulting from the intersection between the rays and the plane $z = 0$ have a purple color.

At the end of this step, we obtain a set of 2D points (more precisely, a set of 3D points with the z coordinate equals to 0). It is more physically correct to follow the light path, and we can consider the result of this ray definition as a consequence of the projection of the light source according to a reference object. Using this way, we can thus consider that we obtain a projection of the reference object shadow on the ground. More precisely, we obtain a set of 2D points on the ground, which determines a shadow.

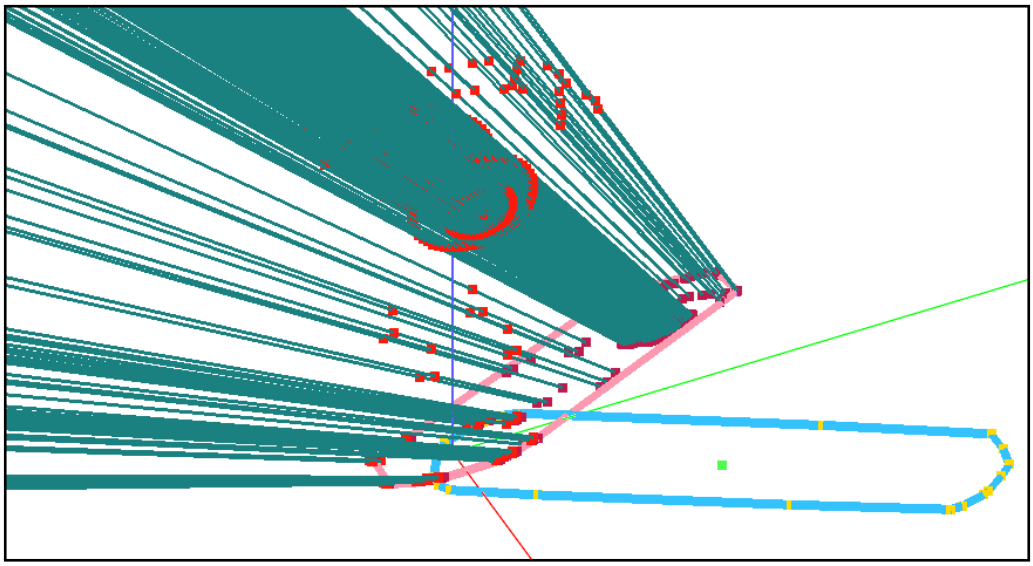


Figure 4.3 – Rays defined using the geometry information (turquoise). The 3D points defined in the previous section are visible in red.

Notes: We can notice that some rays pass through the object. This supernumerary of points is not a problem at this stage. Indeed, because we want to obtain a shadow shape, either a point is necessary to describe it, or it is located inside the shape and does not impact it. Moreover, the ray defines by this 3D point pass through the object for a donate light position. However, because we intend to try several light positions, these rays may change, and do not pass through the object with another environment map orientation.
We also can notice that some 3D points are hidden from the light. For more simplicity and because we cannot define which 3D point is necessary to describe the shadow shape, we compute the intersection for each 3D point. As previously said, a point hidden from the light for a donate environment map orientation, can be visible in another one. Moreover, in this case, it does not impact the final shape.

4.3 Environment map orientation using geometry information of the reference object

The environment map orientation uses the comparison of two shadow shapes. We compare the input shadow shape present in the local scene image (see section 3.2.3) to a computed shadow shape defined by the set of 2D points described in the previous section. We consider that, if the input shadow shape and the computed shadow shape coincide, the environment map orientation is correct.

Our orientation method requires to follow different steps. First, we create a 3D representation of the scene. As previously, we build a 3D representation of the scene using an half-sphere to represent the environment map global scene and the reference object 3D points for the local one. The second step is the shadow shape computation. Then, to compare the two shadow shapes, we compute a metric that computes the difference between them. Finally, we test different light positions, and so different environment map orientation, to minimise the metric and find the correct orientation.

4.3.1 3D representation of the local scene

To recover the orientation of the environment map, we recreate a 3D representation of the scene (see figure 5.5). We first place a half-sphere corresponding to the environment map with a high radius according to the assumption that the environment is at infinity. In our 3D representation of the 360° camera, the lens height is not 0 and is known (see section 3.2.5), we call this height z_{lens}.
Because we know the geometry of the 360° camera and so lens height, we can be precise and not set the environment map center at point $(0, 0, 0)$. Indeed, this point is the center of the basis of the reference object but not of the lens center. We can place the environment map center at point $(0, 0, z_{lens})$, which corresponds to the real environment map center.
This difference is not important compared to the environment map radius but presents some advantages. First, this positioning is more physically realist. Second, even if we do not notice significant result improvement, this positioning can make a difference in cases with low light source position (*i.e.*, close to the ground). In any case, this consideration is inexpensive in the calculation, and there is no additional information to provide.

As said in the previous section, we recover the 3D points of the reference object to obtain the local scene. We recover the 3D light source position from the 2D one using spherical projection formulas (see equations 5.7, 5.8 and 5.9).

4.3.2 Orientation of the environment map using the reference object geometry information

We describe in this section the different steps to proceed the environment map orientation using the geometry of a local scene object.

Two shadows: In one hand, we have described in section 4.2.2 how we obtain a 2D set of points from the light source and the object 3D vertices, called *computed shadow shape*.

In other hand, we obtain a 2D set of points using the reference object shadow, given as input of the method, using the shadow mask (see section 3.2.4), called *input shadow shape*. The input shadow shape is obtain using the assumption that the ground is flat around the object. Indeed, we recover the 3D shadow points position using the same method asf in section 5.2.1.1. Higher a pixel is in the input image, the deeper it is (*i.e.*, more it is far from the camera). Lower it is in the image, closer it is in 3D.

It is not possible to compare directly these two sets of points, which do not contain the same number of points and which do not present the same density of points. Indeed, because the computed shadow is the result of a projection, we cannot predict the distribution of points in the plan.
Conversely, the input shadow is obtained using the shadow mask of the input local scene image. Thus, the point composing this shadow is placed at regular intervals. To be able to compare them, we must obtain similar entities.

To do that, we respectively compute two convex hulls (or convex envelope) from the two sets. A convex hull is a 2D sub-set of points (*i.e.*, included in the starting set) which has the following property: for each point of the set, either the point is in the convex hull set, or the point is placed inside the closed area described by the convex hull. In other words, every point of the initial set is included in the shape defined by the convex hull. As a consequence, the convex hull set is smaller than the initial one. Because we consider fewer points, the computation time is reduced during the next steps. In addition, we can compare the two shapes, describes by the convex hull sets. Finally, by definition, the convex hull avoids the problem of the object basis. Because we obtain the computed shadow shape using every 3D points, the base of the object is included in it. Conversely, because the input shadow shape is obtained using the mask, the basis of the object (and by extension all hidden parts) is not included in it. Using the convex hull allows filling the holes in the input one.

Note: We cannot compare the convex hull sets directly for the same reasons as for the entire set of points. Indeed, even if the input set of points is static because obtained from the shadow mask, the computed set of points may change with the environment map orientation (*i.e.*, the light source position). Thus, the number of points of the calculated shadow convex hull may vary and not be the same than the input set of points.
Moreover, it is not legitimate to compare convex hull sets points by points. Indeed, we have no insurance about the positions of the convex hull points on the plane, and so, we

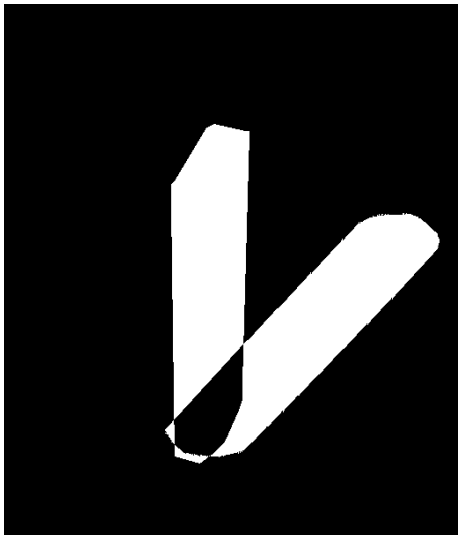


Figure 4.4 – Comparison of the two shadows shapes (top view). The input shadow shape (diagonal one) and the computed shadow shape (vertical one, corresponding of the initial place of the light source).

have no guarantee that points correspond one by one.

The figure 4.4 present the two shadow shape from the top view. The diagonal one is the input shadow shape, and the vertical one is the one we compute corresponding to the initial light position. The light position is arbitrarily placed at a $y = 0$ position (corresponding to the bottom of the image). The *theta* orientation is computed from this starting point. We can notice that the two shadow shape overlap (black part) around the object basis position (*i.e.*, point $(0, 0)$).
At the end of this step, we obtain two binaries images, one for each shadow shape. The two shadow shape images are combined and shown in figure 4.4.

Euclidean Distance Transform: To compare the two shadow shapes, we use the Euclidean Distance Transform (EDT). The EDT image is a gray-level image, which presents on each pixel the distance to the closest pixel of a region of the input image. In our case, we compute the distance, for each pixel in the image, to the closest pixel of the input shadow shape. Because the input shadow does not change, the EDT values have to be computed only once and not at each comparison with the computed shadow shape.

The figure 4.5 shows the result of the EDT. The darker a pixel, the closer it is from the input shadow shape. The pixels of the shadow shape has the value 0. The further a pixel is from the input shadow shape, the higher its value is. In practice, we compute the EDT

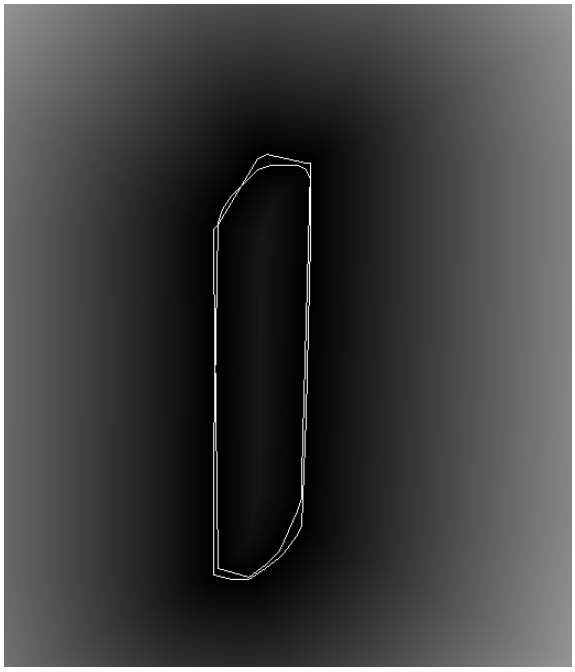


Figure 4.5 – Euclidean Distance Transform of the input shadow shape for the comparison with the computed shadow shape.

on the boundaries of the shapes to obtain non-zero value inside the shape. Otherwise, a computed shadow shape, totally included in the input shadow shape, but different from it, still presents only zero distance values.

We apply the computed shadow mask on this input EDT image. We obtain an image containing zero values everywhere, except for the pixels where the computed shadow shape differs from the input one. Summing these values, we obtain a measurement of the difference between the two shapes. We define this way a metric:

$$M_{EDT} = \sum_{\substack{pixel \\ values}} EDT_{image} * Computed\_shadow_{image} \tag{4.4}$$

This metric renders the similarity between these two shapes or can be seen as the computation as the error of the environment map orientation and scale. By rotating and changing the scale of the environment map we change the light source position (see section 3.2.2). We thus change the rays that we project on the ground floor and so the computed shadow shape. We obtain a different metric value. The higher the metric is, more different the shapes are. The lower the metric is, the more similar the shapes are. If the two shapes are equal, all the computed shadow shape pixels correspond to an input shadow pixel, and so, their EDT value is 0, which implies that the metric is equal to 0.

Comparison of EDT value: The metric 4.4 compares the input shape with the computed shape. Since the shape is computed depending on the light source position (corresponding to an environment map orientation and scale), the metric depends on the light source position. Thus, the metric becomes:

$$M_{EDT}(theta, scale) = \sum_{\substack{pixel \\ values}} EDT_{image} * Computed\_shadow_{image}(theta, scale) \quad (4.5)$$

Using this metric, it is possible to compare two different $(theta, scale)$. Or, in other words, two different couples environment map orientations, to define the one which fits best with the input shadow orientation (*i.e.*, the orientation closer to the real orientation). This metric may be seen as the distance between two shapes. Thus, the couple $(theta, scale)$ which corresponds to the minimum value of the metric, is the one which corresponds to the closer computed shadow shape to the input one.

Minimisation step: To recover the correct environment map orientation, we proceed to a minimisation step. The couple $(theta, scale)$ which minimises the error, presents the maximum similarity between the computed shadow and the input shadow. It is the best environment map orientation which corresponds to the real environment map position. We use a brute force method to recover the correct orientation: we proceed to the computation of the metric (4.5) using 360 possible orientations and different scales. The experience shows that the accuracy of the method is not relevant under the degree. In practice, only a few scales are tested (around 10).
Our method aims to recover the environment map orientation (*i.e.*, the *theta* angle), and the scale result does not have a physical meaning. Indeed, because we consider the environment at infinity, the environment map founded scale does not match with this assumption (because we obtain a finite value). However, the main reason is that, at large scale, the computed shadow modification is not essential: a modification of the scale does not significantly impact the computed shadow shape. The metric does not vary when the environment map scale is huge compared to the scale variation. Thus, if the scale modification is under an epsilon value (and so the computed shadow shape), we stop the best scale search.

Testing every couple allows us to select the best one. The figure 4.6 shows the metric result of a scene for 360 orientations and ten scales. The whiter a pixel is, the higher the metric is. The blacker a pixel is, the lower the metric is. The red pixel is the pixel with

Figure 4.6 – Minimisation step. Minimisation of the EDT metric value between the two shapes. The image present 360 different possible orientations and 10 different scales.

the lowest metric. We can observe that, for this scene presenting a single light source, the metric variation presents a global minimum.

4.4 Results of the method using the geometry information of the reference object

In this part, we show results we obtained using the geometry information of the reference object and how we test it to ensure its accuracy, especially for the method using the 3D model of an object.

We have developed a way to test our method by comparing the relative retrieved environment map orientations. By controlling the relative orientation between the two scenes, we can measure the accuracy of the result and define relative orientation errors.

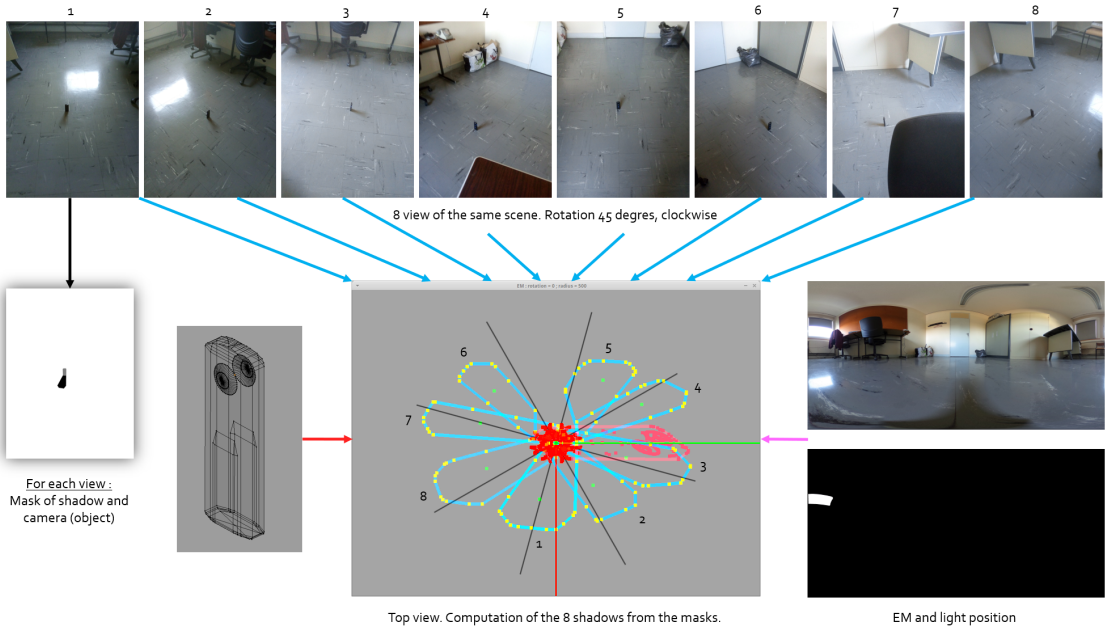


Figure 4.7 – Top line: 8 viewpoints of a same local scene. The environment map (bottom, right) and main object (bottom, left) are the same for all the local scenes. The results of the eight input shadows are shown in the bottom center image (blue), a top view of the local scene. Black lines have been affixed and represent the ground truth (45 degrees angles).

To validate our method, we use a set of eight images of the same local scene taken at different viewpoints. The camera viewpoints rotate with a regular angle of 45 degrees around the reference object (figure 4.7). The results of the eight input shadows (blue) are presented in the bottom center image, a top view of the local scene. To each scene of the top line of the figure (numbered from 1 to 8), correspond a shadow on the bottom center image and so an environment map orientation (with the same number). Because each scene of the top line presents a 45 degrees angle of rotation with the previous or the next scene, the founded orientation of the shadow (*i.e.*, the environment map orientation) must also present a 45 degrees angle of difference. The ground truth is shown in black.

We observe that the eight shadows are oriented in the correct order, from 1 to 8 in the counterclockwise. The scene 1 (figure 4.7, top line, left), presenting a shadow in front of the reference object, presents a computed shadow on the bottom center image of the figure 4.7, aligned with the y-axis, the one toward us in 3D. For the eight scenes, we observe that the computed shadow corresponds with the shadow in the local scene image. However, we observe that some resulting shapes are not well aligned with the ground truth because of missing information when the reference object hides a part of the shadow. Indeed, the local scene 5 and 6, on figure 4.7 on the top of the top view, present an angle larger than 45 degrees. The computed shadow of scene 5 is on the right of the ground truth, and the computed shadow of scene 6 is on the left of the ground truth. That is due to the missing part of the shadow hidden by the object itself, which does not allow to reconstruct a full input shadow. Thus, the computed shadow which results from the 3D object is full and not the same as the input one. We cannot perfectly match the two shapes, and the orientation is not correctly found. Conversely, when the shadow is in front of the main object (figure 4.7 on the bottom of top view), as for the scenes, 1, 2, 3 and 8, the shapes are aligned with the ground truth and present a correct orientation respectively to the scenes around.

scene	expected θ	computed θ	computed η	$error_1$	$error_2$
1	189	189	455	0	14
2	144	140	455	4	4
3	99	106	410	-7	-11
4	54	60	320	-6	1
5	9	26	455	-17	-11
6	-36	-58	455	22	-6
7	-81	-96	230	15	-7
8	-126	-140	455	14	-1

Table 4.1 – Orientation results of the eight local scenes of figure 4.7.

Numerical results are presented in table 4.1. We expect to find exactly 45 degrees between each pair of consecutive images. Each column of the table is (from left to right): the scene number, the expected theta angle, the computed theta angle, the computed environment map radius, and two errors to measure the accuracy of the method. We estimate the overall error using two error metrics (oriented angles). The first one, $error_{1_n}$, uses one scene as a reference and computes the difference between its current angle and the computed scene angle (plus $n * 45$, modulo 360), with n the scene number:

$$error_{1_n} = (angle_1 - angle_n + n * 45) \,\%\, 360 \tag{4.6}$$

The $error_{2_n}$ computes the relative error to the previous scene:

$$error_{2_n} = (angle_{n-1} - angle_n + 45) \,\%\, 360 \tag{4.7}$$

Mean errors are respectively 10.625 and 6.875 degrees. Larger errors are associated with viewpoints 5 and 6 due to incomplete recovery of the input shadow, because behind the object. The maximum error is 22 degrees for the $error_1$ and 11 degrees for $error_2$.

We did not succeed to identify a method we can directly compare with, due to different input data. We can, however, compare our results to a recent method [JSZ+19] which finds the orientation of an environment map using deep learning. In this approach, the average error of the found angle is much higher (around 30 degrees) than our recovery error (10 degrees), and most importantly, they do not consider the environment map scale. We found in our experiments that, although scale does not have a determined physical meaning, it has a clear impact on the results.

4.5 Discussions & limitations identification

In this part, we present the different identified limitations of our method.

4.5.1 Geometry & depth information

During the 3D recovery of the local scene (see sections 4.2.1.1, and 4.2.1.2), the goal is not to obtain a perfect geometry but to obtain a volume to bearing the problem of rays intersection with a 2D impostor (see section 5.5.3) as used in the previous methods.
However, to do that, it is necessary to add the depth or geometry information as input data. That goes against our goal to have a method with fewer input data possible. Our

experience brings us to add it to the input data. Moreover, it is not always possible to obtain it, and all devices do not allow to capture such information. However, it can be considered as not important adding. Indeed, more and more devices allow the user to obtain the depth information as the Kinect (Microsoft), RealSense (Intel), or i-phone X (Apple) cell phone. Moreover, adding the geometry information of a known reference object, and use it for many scenes, does not compel us to recover the geometry for each orientation but only once.

Other problems are linked to the way to obtain 3D information. Even if our goal is not to obtain a perfect geometry, more this one is precise, more the result is. As we can see in figure 4.1 on the bottom, the geometry of the cube is not fully completed. Indeed, the top and the front faces of the cube have a good reconstruction, but the left face is only partially reconstructed. Some grooves appear due to the object orientation vis-a-vis of the camera.

4.5.2 Geometry information of the reference object

Some problems are linked to the capture of the reference object information itself:

- The depth information present in every case, hidden part of the object, and so bring to orientation mistakes as for the scenes 5 and 6 of results presented in section 4.4, figure 4.7.

- The geometry of the reference object can be reused for several scenes if the reference object is the same. But, the depth information of the scene changes in each scene and have to be computed in every case. Thus, the depth is a full additional input, unlike the object geometry.

- Using capture of the depth, we obtain an incomplete geometry of the object. Indeed, we obtain a depth value for all pixels present in the input images (depth and mask images). However, we do not obtain any depth information for parts of the object which do not appear in these images. All parts of the object hidden (occlusion and self-occlusion) are not reconstructed in the final 3D geometry. There are holes in the reconstruction. Because we compute a shadow shape using this 3D information, if this one is incomplete, the shadow shape will be modified and so the result.

The previous inconveniences led us to prefer in practice the geometry of an object than the depth information, to use our method.

4.5.3 Other problems

The presented method uses brute force to test every possible couple (*theta*, *radius*) for the environment map and so for the light source position. But, as seen in figure 4.6, the minimisation procedure has only one global minimum. This figure presents the metric profile for a scene with one light source. A scene with several light sources will present different local minimums for the positions of the light source which provides matches for only a few computed shadows and input shadows. But, even in these cases, a minimum corresponding to the position of the light source which provides matches for every shadow exist.
The computation time to find this position can be improved compared to the brute force method. Indeed, because this problem can be seen as the research of a global minimum, a gradient descent algorithm can be used to test only a part of the solution (*i.e.*, couple (*theta*, *radius*)), and so improve the computation time.

Finally, in some cases, the reference object geometry can be unknown. For example, in a museum where some areas are restricted, or in inaccessible places, the reference object cannot be placed at the center of the scene and cannot be captured into the local scene image. In this case, the user must provide the geometry of an object already present in the local scene. Such an object can be uneasy to models, and some technical skills are necessary to obtain a good result. Moreover, as seen in section 5.5.3, the use of 2D or impostors do not provide good results in every case. The necessity of a known 3D object still is a limitation of our method.

4.6 Conclusion

We presented in this chapter a method using the geometry information of a part of the local scene (*i.e.*, the reference object) to recover the environment map orientation, which uses the 3D of a reference object present in the local scene. We first show in the previous chapter conclusion why the 3D of the local scene object is necessary. This chapter presents two methods to recover the reference object 3D information.
First, a method using a depth map and computing the 3D information by recovering the depth of each pixel of the reference object present in the input image. Second, a method using a 3D model of the reference object to obtain the 3d information. The second reconstruction method has been selected, as discus in section 4.5, because easier to implement and to bring no 3D reconstruction problem (unlike the depth method).

Moreover, this method is more accurate because providing no holes during the object reconstruction step.

To recover the environment map orientation, the method traces rays from the light source present in the environment map to each vertex of the 3D points of the object. Then, points are projected on the ground to obtain a computed shadow. This shadow is compared to the input one which corresponds to the real environment map orientation.
A metric is used to compare the two shadows. By minimising this metric, we can find the computed shadow which corresponds the best to the input shadow, and so to the better environment map orientation.

We present results and expose how we test our method accuracy by comparing the relative founded environment map orientations compared to expected ones. Our method presents mean errors about 11 degrees. We think that the accuracy of the method can be improved, and discuss that in the next section.

4.7 Possible improvements

In this part, we talk about the different improvements which can be implemented easily in future work. These improvements may allow us to obtain better results and to obtain a more realistic model:

Several light sources: Taking into account several light sources may allow our method to be more robust to different scenes. Indeed, currently, the method considers the main light source of the scene. But, considering the full light profile, and so the entire information provided by the environment map may allow being more robust in cases where no main light source is present. For example, if the light profile change in a pipeline providing a 360 degrees video, in a vehicle orientation in an environment (see section 2.6.2) method.

Light source specificity: Taking into account the specificities of the light sources (colors, intensities, *etc.*) can also allow our method to be more robust. Indeed, if different light sources are present in the environment map, the specificities of each light source provide useful information for the orientation. For example, in a case presenting three lights, a green one, a yellow one and a red one, with different sizes, the three resulting shadows colors and shapes give us useful information to decide which light source is linked with which shadow, and so an *a priori* environment map orientation.

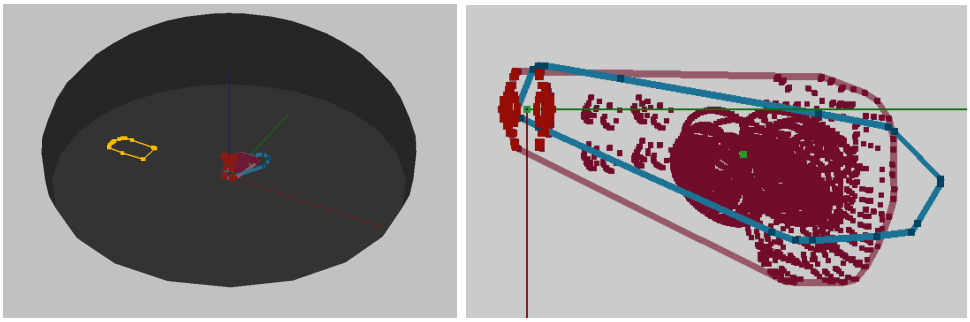


Figure 4.8 – Area light source. Left: global scene. The area light source is represented in yellow by a set of points. Right: top view of the local scene and effect on the computed shadow shape (in pink) compared to the input one (in blue). The computed shadow shape is larger than the input shadow shape and cannot be compared to it.

Take into account area light sources: Another possible improvement of our method is to take into account area light sources. Indeed, in the current method, we consider the light source as a point. Every ray is projected from this point, and so, the resulting shadow shape depends on it. But, on the environment map (and in real-world), the light source is represented by an area. Project rays from every point of the light source change the resulting shadow shape. Considering such shape may correspond more to the reality than consider a unique point.

Our first tests show too large surfaces (see figure 4.8) to obtain similar shadow shapes, and so expect better results. That is due to the 3D position of the light source in our representation of the scene. Indeed, the light source is closer than in the real scene.

Environment map vertical orientation: One of the assumptions is to consider the environment map as already well oriented horizontally. We are just looking for the *theta* angle (see figure 3.1). However, because of the capture device, the ground, or the environment map image reconstruction method, the horizontal orientation is not always ensured. Taking into account the environment map orientation (*theta* and *phi* angles) would make the model more realistic and allow us to have more accurate results.

Special cases: Taking into account special cases or avoid them. Indeed, for example, when the light source is placed behind the camera which takes the local scene, the reference object shadow will be hidden by the reference object (*i.e.*, self-occlusion).
The matches (section 3.3, light path 1) find during the ray-definition step (finding matches between object and shadow boundaries points) are not coherent.
In other cases (section 3.3, light path 3) the comparison between the input shadow hidden by the reference object and the computed one (and so, fully complete because computed

from the model) cannot be well operated.

Several reference objects: Finally, taking into account several reference objects may allow us to find by triangulation (using shadow comparison) the light source position in a better way than using the scale value of the environment map. As already discussed, the founded scale value has no physical meaning in our method, although necessary to find similarities between the shadow shapes. But, taking into account the real light distance from the local scene, may allow obtaining better computed shadow and so to improve results.

Chapter 5

Reducing hypothesis for automatic orientation of an environment map around a local scene

Résumé

Dans ce chapitre, nous étudions des solutions afin e réduire les hypothèse de la méthode d'orientation présentée dans le chapitre précédant. Nous nous intéressons à deux approches mises en place pour orienter automatiquement une carte d'environnement par rapport à une scène locale. Ainsi des méthode basées uniquement sur des informations 2D, utilisant des information 2D dans un environnement 3D sont proposées. Ainsi, les données d'entrée 3D ne sont pas utilisées ou uniquement partiellement. Ces essais ont été réalisés dans le but de trouver une méthode partant de données minimales (*i.e.*, seulement les images de la scène locale et de la carte d'environnement). Un transfert à la 3D est alors par la suite opéré par projection. Deux méthodes de passage à la 3D sont considérées dans ce chapitre :

- La première présente une définition complète des rayons en 2D, transformés ensuite en 3D.
- La seconde présente une représentation de la scène par imposteurs (représentation 2D d'un objet dans un environnement 3D afin de simuler sa présence sans en connaitre la géométrie) pour directement obtenir les coordonnées des rayons en 3D.

Deux approches de positionnement de la carte d'environnement sont alors proposées, chacune partant de la représentation des rayons extraite des données 2D.
Nous exposons, par nos résultats, les limites d'une telle représentation et la necessité d'informations en 3D pour obtenir plus de robustesse lors du calcul de la solution.

Definition des rayons : Nous définissons dans un premier temps les rayons de deux manières différentes. La première manière définit les rayons de lumières en 2D entre l'objet de référence et son ombre. Cette approche a l'avantage de ne nécessiter que l'image de la scène locale pour définir les rayons. Elle présente par contre plusieurs désavantages : les rayons doivent être définis de l'ombre à l'objet de référence et ne prennent pas en compte la position de la lumière. Les rayons sont alors statiques et ne dépendent pas de la position de la lumière de la carte d'environnement. La mise en correspondance des rayons et de la source lumineuse n'est alors pas assurée. Une étape de récupération de la 3D est ensuite effectuée.
La seconde manière de définir les rayons est directement en 3D, en utilisant des imposteurs afin de résoudre les inconvénients cités précédemment. Un imposteur est utilisé pour représenter l'objet de référence, un autre l'est pour représenter l'ombre correspondante. Ces imposteurs représentent la scène locale. De cette manière, les rayons peuvent être définis entre l'ombre et la source lumineuse.

Orientation de la carte d'environnement : En relation avec les manières de définir les rayons, deux méthodes d'orientation de la carte d'environnement ont été testées. La première, utilisant la définition des rayons 2D, consiste à placer la source lumineuse présente dans la carte d'environnement dans le cône de rayons. La seconde, utilisant les imposteurs, consiste à placer l'objet de référence dans le cône de rayons définis entre l'ombre et la lumière en orientant correctement la carte d'environnement.
Dans les deux cas, nous considérons que la carte d'environnement est correctement orientée si la source lumineuse, l'objet et son ombre sont alignés.

Résultats et limites : Nous présentons dans la section suivante les résultats de ces deux approches. Plusieurs cas présentent une orientation visuelle correcte de la carte d'environnement. Seulement, nous avons identifié de nombreux cas où une orientation correcte n'est pas possible ou seulement partielle (*e.g.*, cas où la lumière est positionnée derière l'objet de la scène locale, trop basse ou cas présentant plusieurs sources lumineuses).

Conclusion : La dernière partie de ce chapitre est dédiée à une conclusion de ces méthodes et aux enseignements que nous pouvons en tirer. En effet, la faible robustesse et le manque de précision des résultats présentés dans ce chapitre, montrent l'importance de la 3D réelle de la scène locale ou d'une partie de celle-ci pour orienter correctement la carte d'environnement.

5.1 Introduction

In this chapter, we explore different directions to reduce hypotheses linked to our method presented in chapter 4 to orient an environment map automatically. Approaches presented in this chapter are 2D orientation methods. The 3D input data is not used in the methods presented below. However, to recover the orientation (*i.e.*, registration) of the environment map (in 3D), a 3D recovery step is necessary. Two methods are defined here:

- The first one presents a full 2D rays definition, and then, a 3D recovery of the rays from the 2D ones.
- The second one presents a partial 2D rays definition, using impostors, to directly recover a 3D set of rays. An impostor is an image which represents an object in a 3D environment to simulate it without any 3D knowledge.

Thus, we first present the rays definition in detail. Then, we present the orientation step of the environment map using the set of rays and present different results. Finally, we discuss results and expose the different limitations of these approaches to explain the necessity of the 3D of the local scene to obtain a correct set of rays. In other words, we can see these approaches as preliminary ones and as a persuasion of the necessity of the 3D of the local scene.

5.2 Ray definition

We define here the process to set up rays between the reference object and its shadow. We cannot directly obtain a correct set of rays in the direction of the light source. We first have to determine which part of the object is linked to which part of the shadow.

5.2.1 Generating 2D rays

We present in this section the chosen approach to define 2D rays from a 2D image that describes the light direction. The main idea is to find correspondence (*i.e.*, matches) between points of the shadow and points of the object of the input image (see figure 3.3, right) and to project them into the direction of the light. The intersection of the 2D cone with the environment map representation defines the 2D area of the light source. Placing the light source in this area gives us the orientation of the environment map.
We rely on the orientation detection of a shadow projection on a ground plane according to the object of this shadow, in the image plane. The ground plane is then considered as flat

and horizontal. By getting matches between points of the shadows and points of the object, we can define an approximate area for the light source in the image plane. Subsequently, we improve the results by removing the wrong matches. We seek to find the precise position of the light source on a half-sphere. We perform this process for each luminous (shadow/source) pair, we search for the optimal orientation of the environment map which is the most compatible match between the light source position and the projected area on the environment map.

In this section, we describe how we create the rays, and then explain how we remove false positives to keep only the relevant rays. Finally, we will see how we find the depth of the rays and how we project them on the representation of the environment map to find its orientation. The first step is to get matches between points of the shadow boundary and points of the silhouette of the reference object.

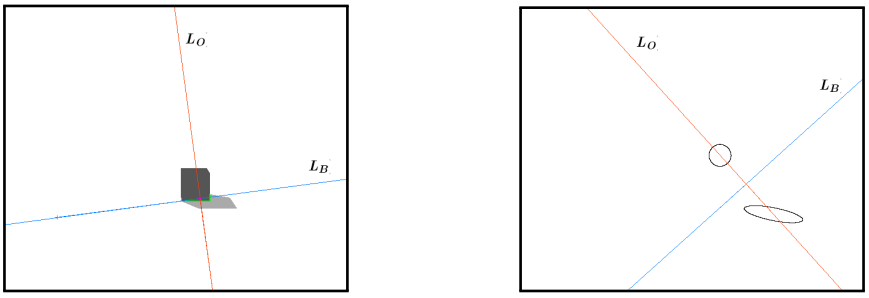


Figure 5.1 – Lines computation using shadow and object masks. The blue line separates the two object. The orange line separates the left and right sides of the masks. Left case with a common border (green). Rigth case without common border.

Define separation 2D lines: To find relevant points and matches, we have first defined two lines computed from the object and the shadow masks: the first ($\boldsymbol{L_B}$) is a separation line between the two masks (see figure 5.1, blue lines). There are two cases.
In cases presenting a common border between the object and the shadow (see figure 5.1, left, green pixels), we compute this line as the one which minimises the sum of the distances to the border (green pixels).
In cases with no common border (see figure 5.1, right) we compute this line as perpendicular to the second one ($\boldsymbol{L_O}$, orange lines in figure 5.1), passing at equal distances of the center of the bounding boxes of the masks.
The second line ($\boldsymbol{L_O}$), passes by the center of the two bounding boxes of the masks in the case without common border. This line can also be defined as perpendicular to $\boldsymbol{L_B}$ passing

by the center of the border (green in figure 5.1) in the other case.

Using this way to define lines $\boldsymbol{L_B}$ and $\boldsymbol{L_O}$, in every case, the two lines are perpendicular. The $\boldsymbol{L_B}$ line define two areas on the image, one for the object, one for the shadow, and the $\boldsymbol{L_O}$ line defines two areas in the image, one for the right size of the object and shadow and one for the left size.

We define matches as a couple of pixels: one from the object, one from the shadow. The goal is here to define parallel lines using matches. To do that we want to ensure that pixels of a match are selected from the same side of the shadow and object. We define matches between shadow and object points (*i.e.*, pixels) using the above define lines and the following 2D criteria:

- For two pixels of a same match, their respective distance is bounded by the size of the area of the object mask and the shadow mask. Matches at close distance to $\boldsymbol{L_B}$ are removed.
- A point of a match cannot be too close to $\boldsymbol{L_B}$. Indeed, at a small distance, a small error (of few pixels) for a match involves a big error for the direction of the match.
- A match line cannot cross $\boldsymbol{L_O}$. Indeed, we suppose that a pixel of the left (respectively, right) part of the shadow matches with a pixel of the left part (respectively, right) of the object. The goal is to have the smallest cone angle possible for the direction of the light.
- If a match line crosses too many other match lines, we do not keep it because the match direction is in contradiction with most matches. It helps to keep a closer bounding area of the light source.

The values of criteria parameters can be modified. For example, the difference of distance of the first one can change (10% or 15% of difference), the distance to $\boldsymbol{L_B}$ can change (depending on the size of the object) for the second or the number of crossing matches can change (we have defined that 10% of the total of matches is a correct value). The matches are defined taking a shadow point and an object point randomly, and verify if the criteria are respected.

The last step is to prolong the matches to obtain a cone of rays, to locate the light source, as seen in figure 5.2. The left and middle columns show two results for the same local scene with a different value of the criteria's parameters. The number of matches and the angle of the cone are different. On the left column, the matches are shown in red, and

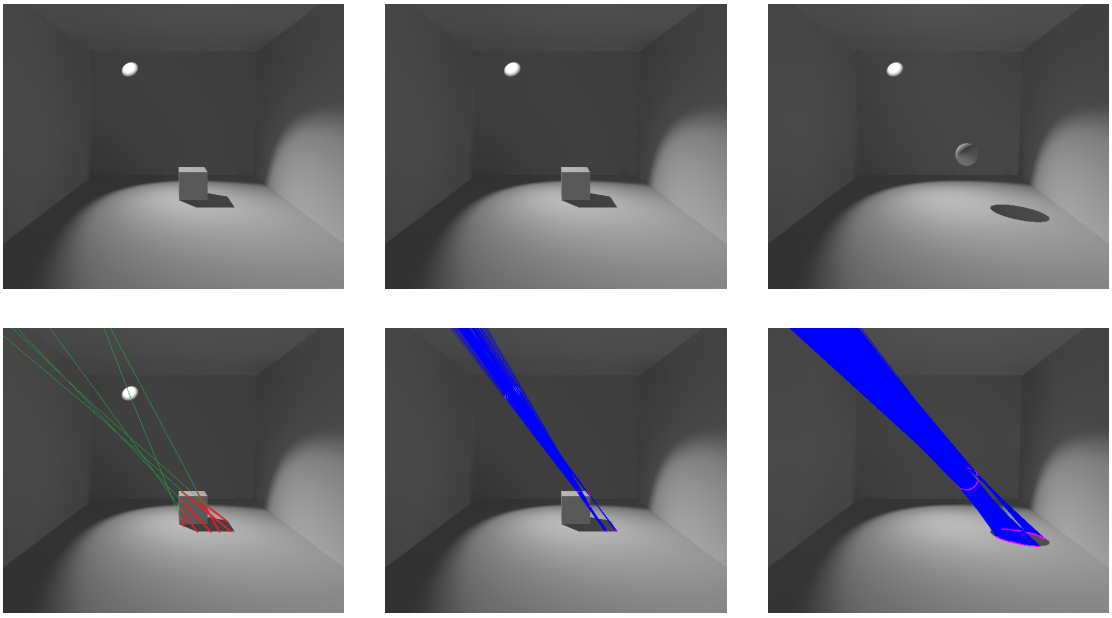


Figure 5.2 – Result of the matches of three scenes: object in contact with the ground (left and middle columns) and object without any contact with the ground (right column). On the bottom left image, only a few rays (green) are shown. The matches between the reference object and its shadow are shown in red. The top row presents input data. The bottom row presents matches.

the extension of the matches in the direction of the light has been highlighted in green. The two other columns show matches in blue.

5.2.1.1 Generating 2D rays and their transformation to 3D

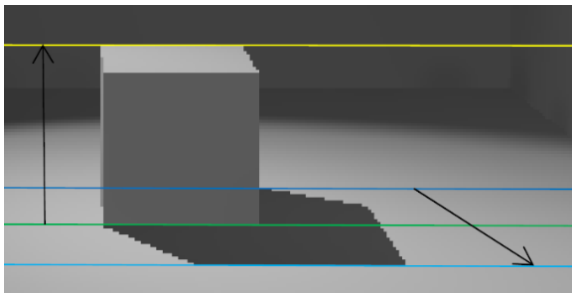


Figure 5.3 – Conversion of each match from 2D to 3D.

We convert matches' coordinates from 2D to 3D by making two simple assumptions. The first one is that the shadow is projected on a flat ground (*i.e.*, on a plane). The second one is that the object is also planar, which is, of course, incorrect and a coarse approximation. The figure 5.3 shows how we convert to 3D (x, y and z value) the 2D coordinates of each point (shadow and object) of each match:

- For a pixel of the shadow, we consider that we know x equals to x_i, the x value of the pixel in the image, minus x_0, the x value of the middle of the object mask (because we consider the middle of the object as the center of the local scene), and $z = 0$ coordinates (because the shadow lays on a flat ground).

  To recover the y (*i.e.*, depth) value, we consider that the higher a pixel is in the image, the deeper it is (*i.e.*, further it is from the camera). To find the y value, we subtract y_p, the y value of the pixel (between the two blue lines in the figure 5.3) to y_c, the y value of the center of the object (y value of the green line). Using that, we obtain a negative value for the higher pixel in the image (*i.e.*, deeper) and positive values for the other ones.

$$x = x_i - x_0 \tag{5.1}$$

$$y = y_c - y_p \tag{5.2}$$

$$z = 0 \tag{5.3}$$

- For a pixel of the object, we consider that x and z values of a point are respectively equal to the x_i and y_i value of a pixel in the image. Since we assume that the object is on a plane, we can consider that the higher a pixel in the image, the higher it is in 3D. We recover the y value of an object point by subtracting y_i, the y value of the pixel in the image (between green and yellow line in figure 5.3) with y_0, the y value of the base of the object (green line). Using that, we obtain a positive value for all the points of the object.

$$x = x_i \tag{5.4}$$

$$y = y_i - y_0 \tag{5.5}$$

$$z = y_i \tag{5.6}$$

At the end of this step, we consider that we obtain a set of rays which describes a 3D cone. The intersection of the cone and the environment map describes an area that is the location of the light source.

5.2.2 3D rays' generation from an impostor representation

This method directly defines rays from the light source present in the environment map to the shadow of the object present in the local scene. To orient the environment map, the object of the local scene boundaries has to be aligned with the rays.

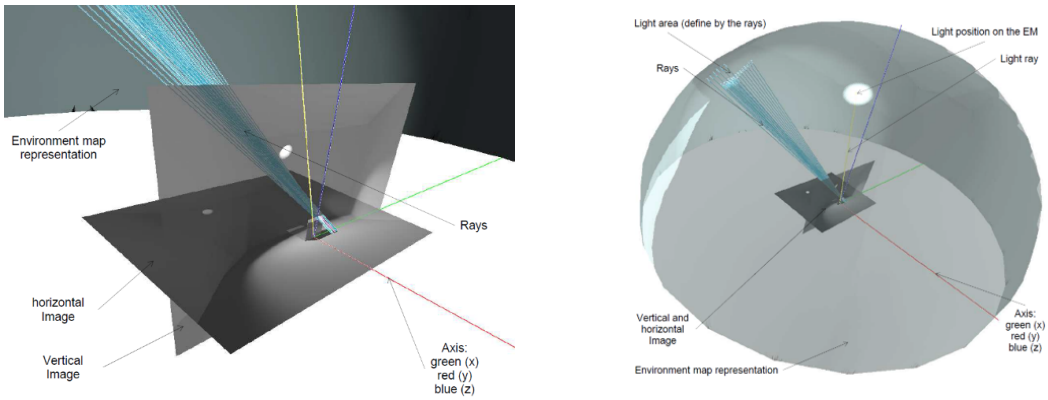


Figure 5.4 – Creation of the 3D representation of the scene for the second tested method. Left: impostors used for the local scene. Right: global representation of the scene.

The first step of this second method is to create the 3D representation of the scene. As previously, we place a representation of the environment map defined by a half-sphere (see figure 5.4, right) around a local scene. The local scene is represented by two *impostors* (see figure 5.4, left). An impostor is defined as a 2D representation (*i.e.*, an image) of a 3D object. Here, we use it to represent the reference object and its shadow. We describe in detail how we create the 3D scene using impostors in the next section, 5.3.1. Using this representation of the scene, we define a set of rays directly in 3D, from the shadow points on the shadow impostor to the light source (see figure 5.13, orange rays). Rays start from the boundaries of the light source (represented by an area on the environment map surface) to the boundaries of the shadow representation on the horizontal impostor. The intersection of this set of rays with the vertical impostor (*i.e.*, the representation of the reference object) defines a surface shape (see figure 5.13, left). Our goal is to make this shape coincides with the reference object shape.

The first ascertainment we do is that the obtained area on the vertical impostor should have a shape similar to the object. Solving our problem is solving the orientation of the object shape with the intersected shape on the impostor. Because we do not follow the light path, this ascertainment is not trivial.
Because we do not consider the depth of the object, we make a second ascertainment: rays associating points in shadow close to the object lack robustness and must be discarded.

5.3 Environment map orientation

In this section, we describe the environment map orientation step for each configuration listed in section 5.2. The basic idea is similar (rotate the environment map around the local scene), but some adjustments are made for each configuration.
We first describe how we create a 3D representation of the scene using impostors in section 5.3.1. We then present the orientation procedure for the cases presented in sections 5.2.1 and 5.2.2.

5.3.1 3D representation of the scene from 2D information: impostor

To recover the orientation of the environment map, we recreate a 3D representation of the scene (see figure 5.4). We first place a half-sphere corresponding to the environment map. The center of the half-sphere is on $(0, 0, 0)$ and a high radius, according to the assumption that the environment is at the infinity (the radius of the environment map has been reduced in the figure 5.4 for more readability).
At the center of the scene, we place two impostors representing in the local scene, the reference object and its shadow. The principal advantage is that we already own a 2D representation: the input local scene image. We do not have to apply any process to obtain the representation of the local scene.
Such representation is a correct representation for the reference object shadow because the shadow is projected on the ground. Thus, on a 2D plane (the ground can present some level variation, but we assume that the ground is flat around the reference object). Therefore, the impostor representing the shadow is a fair representation, except that we must take into account its projection on the camera plane. In practice, because we do not want to have any assumption about the camera used to obtain the input data scene image, we do not take into account this projection.
Conversely, the impostor representation is not a correct representation for the reference object because it does not take into account the depth of the object. Because the object is projected on a plane, its 2D shape is not directly comparable with the 2D shape of the shadow.

The two impostors are placed at the center of the scene and perpendicular to each other. The shadow impostor is placed horizontally. The object impostor is placed vertically. The center of the intersection between the reference object plane and its shadow plane is placed at the center of the scene, which also coincides with the center of the environment map (point $(0, 0, 0)$ according to the green line on figure 5.3). The two impostors are superimposed and intersect each other. On figure 5.4, on the left view, is presented an

example of impostors intersection in a 3D scene, and on figure 5.5, left, we can see a zoom on the view of the center of the scene and how the reference object and its shadow are placed.

The matches' coordinates, previously computed in 3D, start from the horizontal impostor to the vertically one and are extended in 3D on the environment map representation. The intersection of all the matches with the environment map 3D representation describes an area (visible on the figure 5.4, right).

5.3.2 Orientation of the environment map using 2D information

The environment map orientation using 2D information includes two steps. Its goal is to find the correct environment map orientation, *i.e.*, by replacing the light source in the set of rays area. First, we find the correct latitude of the environment map by modifying its scale. The second step is to rotate the environment map to modify its longitude and find the correct position of the main light source regarding the set of rays. The two steps are shown in figure 5.6.

Environment map scale: We first find the correct latitude for the set of rays by changing the environment map scale. Modify the environment map scale allows us to move upwards or downwards the area location, relatively to the light source position. Indeed, because we use the light path from the shadow to the object (section 3.3, light path 1), the set of rays is static and does not vary when we apply a rotation to the environment map. Only a change of the environment map scale modifies the relative position of the

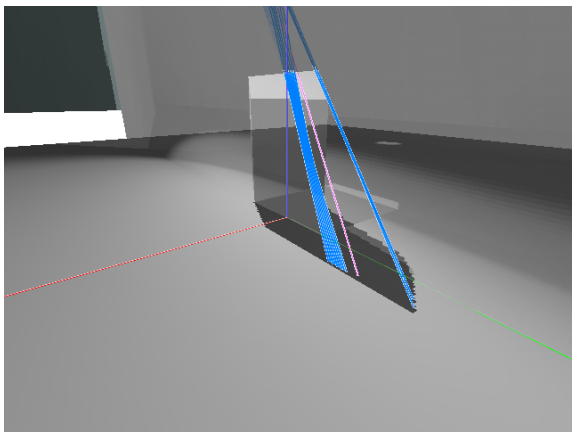


Figure 5.5 – 3D representation of the local scene using impostors. The blue rays are defined using the path from the shadow to the reference object, using light path 1, see section 3.3.

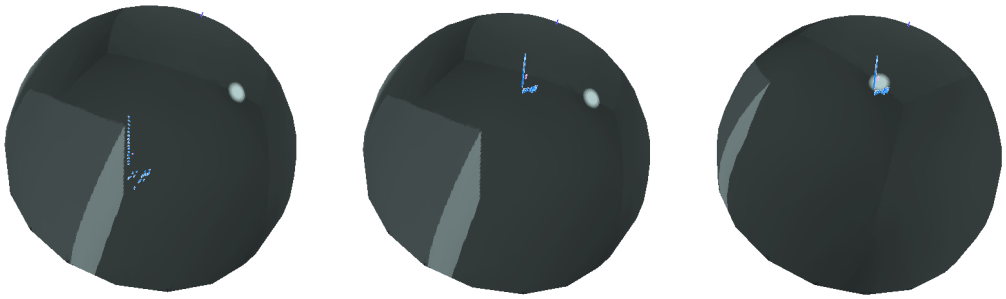


Figure 5.6 – The different steps of the environment map orientation around a local scene using the tested method presented in section 5.2.1. Left: initial position of the environment map for a scene. Center: vertical orientation of the environment map changing its scale. Right: final result of the placement of the light source (white circle on the environment map surface) inside the area described by the rays. The environment map steps are shown using the same size to focus on the rays positions changes.

light source compared to the set of rays latitude. Because we know the position (i, j) of the light source in the input environment map image, which corresponds in an environment map image to a couple $(theta, phi)$, we can compute the 3D position $p1 = (x1, y1, z1)$ using the spherical projection formulas (with r the radius of the sphere):

$$x1 = r * sin(phi) * cos(theta) \tag{5.7}$$

$$y1 = r * sin(phi) * sin(theta) \tag{5.8}$$

$$z1 = r * cos(phi) \tag{5.9}$$

The figure 5.7 shows the environment map light source repositioning at a correct altitude (*i.e.*, the same than the intersection of the environment map with the rays). The top box shows the two different cases: on the top line, the light position is lower than the environment map intersection with the rays. Increasing the environment map scale allows placing the light source at the correct latitude. On the bottom line, the light position is upper than the intersection. Decreasing the environment map scale allows placing the light source at the correct latitude.

The bottom box shows why it always exists a solution to get the correct latitude by changing the scale (point $P1$ if the light source is upper than the rays, and point $P2$ if the light source is lower than the rays). The reason is that rays do not start from the environment map center, but from the shadow boundaries (point R on the scheme). In the scheme, the R position respectively to the center of the environment map (point O) has been exaggerated to present reasonable scales. At the end of this step, we have placed the

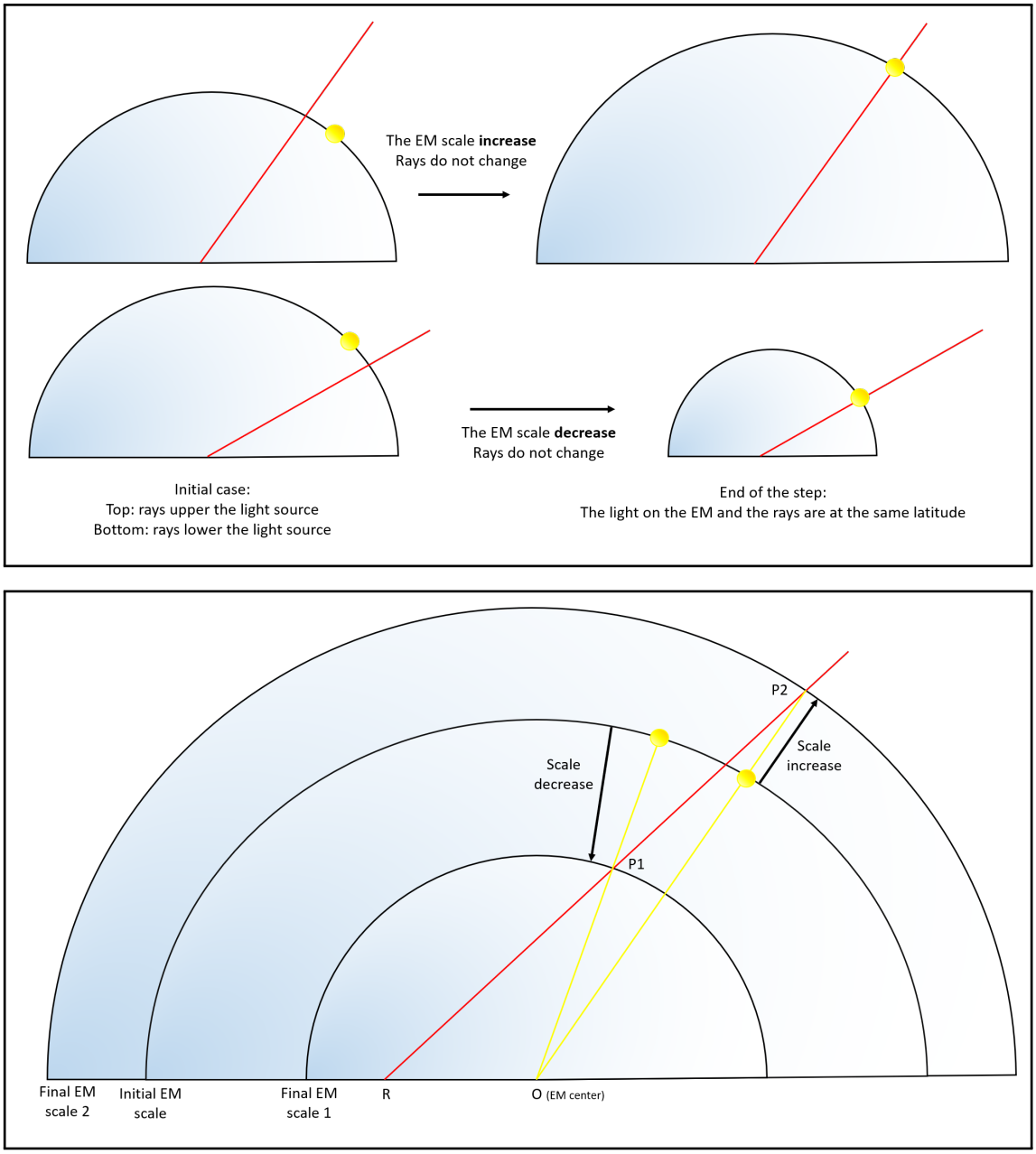


Figure 5.7 – Top box: presentation of the repositioning of the environment map light source by changing the environment map scale. Bottom box: illustration of the environment map scale changing direction in the two different cases. The relative latitude changes because the rays start from R, the shadow boundaries, and not from the environment map center (point O). It exists, in each case, an environment map scale to cross the rays with the environment map at the correct latitude (point $P1$ and $P2$).

light source at the same latitude as the rays (case of the figure 5.6, center).

Environment map rotation: We rotate the environment map around the y-axis which allows us to place the light source and the area on the same plan vertically (*i.e.*, the correct longitude). With that, we can replace the light source inside the area defined by the rays and obtain the orientation of the environment map (see figure 5.6).
Then, we consider the centroid of the environment map intersection with the rays. We obtain a 3D point $p2 = (x2, y2, z2)$. We also consider the center of the light source ($p1 = (x1, y1, z1)$). To obtain the *theta* angle (see figure 3.1), and so the environment map orientation, we have to compute the oriented angle between the vectors $((0,0),(x1,z1))$ and $((0,0),(x2,z2))$. The obtained angle gives us the correct rotation to apply to the environment map (see figure 5.6, right).

5.3.3 Orientation of the environment map using rays generated with impostor

The orientation has the same steps (scaling and rotation steps) as the method using only 2D information to define rays, but for different purposes. In this method, the aim is to place the reference object shape into the shape defined by the intersection of the rays with the reference object impostor (called here, "*target shape*"). If the object shape matches the target rays, the environment map is at the correct orientation. To define if the two shapes match, we have defined a metric and aim to minimise it. When rotating and scaling the environment map, the target shape, and thus, the metric will change.

Environment map rotation & scale: It is not intended to recover the correct latitude and longitude of the light source, rather to adjust a coherent positioning. It is also important to have in mind that during the orientation process, rays need to be computed at each new tested rotation and scale.

We test 180 possible orientations and different scales. We do not have to test 360 possible orientations because the reference object shadow is placed on one side or the other of the reference object impostor. In half of the cases, when the light source is on the same side the shadow, rays do not cross the impostor. There is no intersection and so, no target shape.
The number of tested scales is not fixed. Only a minimum environment map size has been fixed (arbitrarily, ten times the reference object size). Because the environment map is considered representing the far geometry, a maximum size cannot be fixed, and since the

metric minimisation gives better results, the environment map radius can increase. Using that, we obtain a set of rays defining a target shape close to the reference object one.

Metric computation: To determine if the environment map is at the correct position, we compute a metric (called here Msl, metric shadow/light). The metric is defined as the sum of the minimum distance of each ray (of the target shape) with the reference object boundaries.

$$Msl(rotation, scale) = \sum_{\substack{ray \\ intersection}} min\_dist(_{intersection}^{ray}, _{boundaries}^{object}) \tag{5.10}$$

If rays are far from the object boundaries, the minimum distance and the metric will be large. Conversely, if the rays are close to the object boundaries, the minimum distance and the metric will be small. Thus, minimising this metric allows us to obtain a target shape the closer possible to the reference object boundaries. We compute the minimum distance with the reference object because, using the light path from the shadow to the light source does not allow us to obtain correct matches between the shadow and the object (*i.e.*, we cannot define which point of the shadow matches with which point of the object). Using the minimum distance for all rays is not physically correct but ensures to get a correct approximation of the shapes' similarity.

In some cases, the rays intersect the object. That especially happens for rays starting from points of the shadow shape close to the basis of the object. In a large number of light source positions, these rays will intersect the object. Thus, we use the object boundaries and not the totality of the object points, for two reasons.
The first reason is that we use shadow boundaries to define the rays. It is more physically correct to use the object boundaries to compare same units. Moreover, because the object and shadow have approximatively the same size in the local scene input image, we can consider that the reference object boundaries and the reference object shadow boundaries have approximatively the same number of pixels.
The second reason is that if we consider the totality of the object shape, a ray that intersects the object will have a distance of 0. In this case, to minimise the metric, it is preferable to obtain the maximum rays which intersect the object. This situation appears when the environment map radius decreases. Indeed, if the environment map radius decreases, the light source will be closer to the shadows and so, the cone defined by the rays and the target shape, smaller. From this, it results that more rays can intersect the interior of the object. To avoid that, considering the object boundaries, allows obtaining a minimum

distance different to 0 also in cases of a ray intersecting the object.

Metric minimisation: Minimising the metric allows us to obtain a target shape matching best the object shape. If the object is perfectly included in the target shape, then the light source in the environment map is in the correct position to recreate the shadow from the reference object we consider. In this case, we can consider that the environment map orientation is the correct one.
To minimize the metric, we test 180 possible orientations and different scales. We do not limit the size of the environment map radius because it represents the environment at infinity. In practice, we cannot give an infinite radius to the environment map. We stop the minimisation procedure when the Msl difference between the two steps is too small. Indeed, when the environment map radius increases, the cone defined by the rays (and so, the target shape) enlarges. However, when the environment map radius is high, an enlargement of the radius does not significantly change the cone and therefore, the resulting Msl metric will not significantly change either. Because our goal is to find the environment map orientation and because such computation of the scale is not physically correct, in this case, we stop the minimisation procedure. At the end of this step, we obtain the environment map orientation. In section 5.4, we will show the results we obtain, and limitations of both methods described.

5.4 Results

In this section, we show the results we obtain for the method using 2D rays definition and the methods using impostors rays' definition, and describe how the methods performs in different cases.

Rays' definition using 2D information : The figures 5.2 and 5.8 present respectively three and two different results of the matching between reference object and its shadow. We can see that the results are different for all the presented scenes. In the presented results, the light source is visible for a better understanding. However, in a real case, the light source may not be visible in the local scene image. It is not a problem because the ray definition step aims to find a cone that allows defining an area on the environment map surface and not to recover the light position directly. However, if rays surround the light source position in 2D in the local scene image, the area on the environment map surface will be at the correct position to find the correct orientation.
In figure 5.2, are presented from left to right, a case with a cone not precise enough, a case

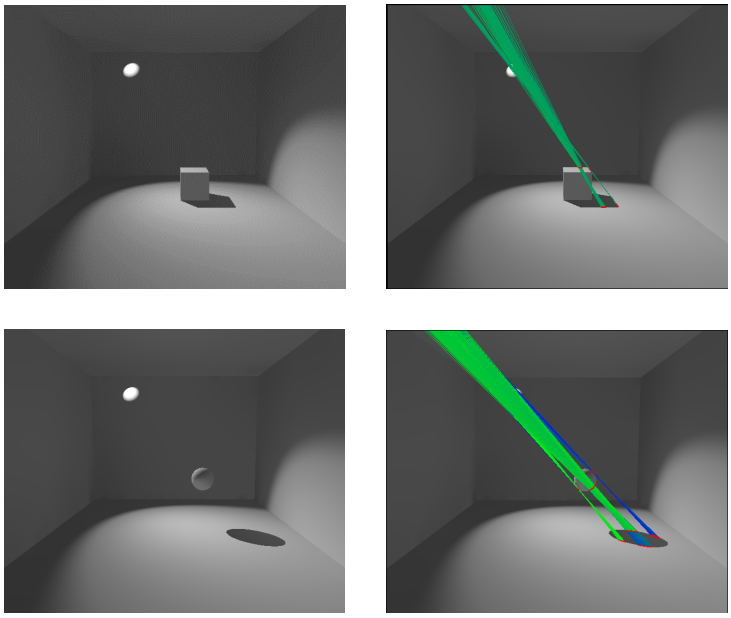


Figure 5.8 – Result of the orientation of the environment map using the 2D rays' definition on different scenes. The light path used is the light path from the shadow to the reference object (see section 3.3, light path 1). Top line: using a cube presenting a common boundary with the shadow. Bottom line: using a sphere without a common boundary.

where the cone perfectly surrounds the light source position, and a case where the cone surrounds the light source position but is very large and necessitates some refinement to be able to find a correct environment map orientation. Figure 5.8 presents a case on the top row, where the cone partially surrounds the light source and a correct case on the bottom row, where the rays are more or less parallel. On this figure, the greener a light ray is, the smaller the distance between the two points of the match is and, the bluer a ray, the higher the distance is. The two points of a match are shown in red.
This result variation is due to the large number of criteria (see section 5.2.1) and parameters to define the cone of rays. Each variation of a criterion can change the resulting cone. Due to the methods using only 2D information orientation result accuracy, we did not try to define in detail which parameters influence the results the most.

Orientation results of the method using only 2D information to define rays:
The environment map orientation is considered here as correct if the light source on the environment map is placed inside the area defined by the rays on the environment map surface. Figures 5.9, 5.10 and 5.11 present orientation results of the method using 2D rays' definition in different cases. In each case, the exposed orientation is the best we have found. Tests have been conducted mostly with subjective assessments. It did not appear relevant to develop an objective method to evaluate the results because we estimated that

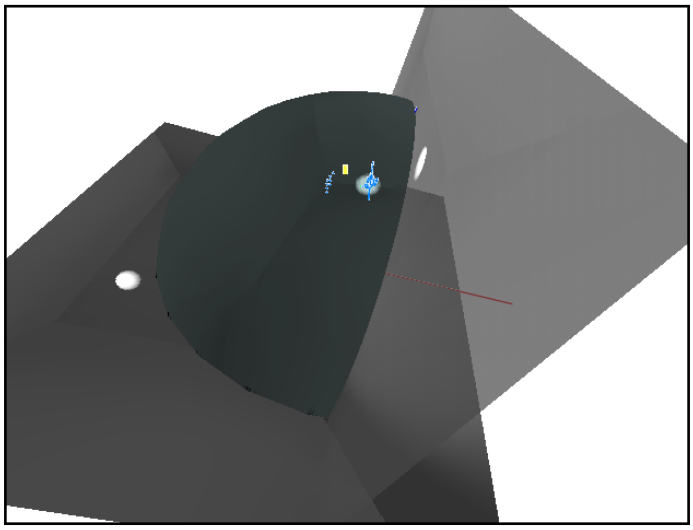


Figure 5.9 – Result of the orientation of the environment map using the 2D ray's definition on a scene, zoom on the local scene. The light path used is the light path from the shadow to the reference object (see section 3.3, light path 1). However, we can see that the environment map radius is too small compared to the local scene (impostors).

the approach was not sufficiently valuable.

Figure 5.9 presents an orientation result of the environment map around a local scene. The light source is correctly positioned within the area defined by the rays (in blue). To obtain a correct result (*i.e.*, obtain the same latitude for rays and light), following the principle exposed in figure 5.7, the environment map radius has been reduced. As a consequence, the environment map and the local scene have the same magnitude.
Figure 5.10 presents orientation with different light profiles. On the left part of the figure, the orientation of a single light source is computed, and on the right part, this is done for three different light sources. We can observe that the orientation for a single light source is correctly recovered. In this case, the area defined by the rays defines a circle which is the same shape as the object, and the light source is perfectly positioned inside. The orientation of three different light sources is optionally recovered with each light source positioned close to that projected position.

Figure 5.11 exposes a special case where the light source, the reference object and the camera are aligned (in this order). In this case, the shadow is placed between the reference object and the camera. We can notice that we do not expose special cases where the light source is placed between the reference object and the camera. Indeed, in this case, the shadow is hidden or partially hidden by the reference object, and no match can be found.

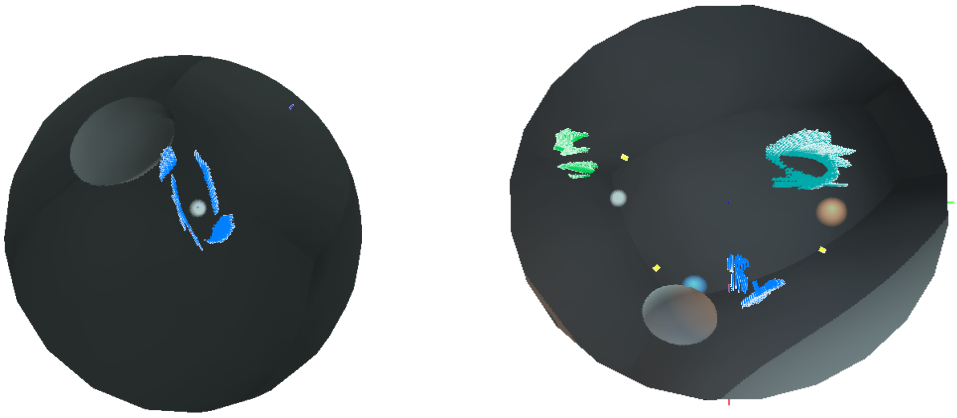


Figure 5.10 – Result of the orientation of the environment map using the 2D information for rays' definition using different lights profiles. The used light path is the light path from the shadow to the reference object (see section 3.3, light path 1). Left: final result of the placement of the light source inside the area described by the rays for one light source. Right: same, for three lights sources.

On the top line, is presented a scene using a cube sharing a common boundary with the shadow. The light has been placed close to the zenith of the cube to obtain a small shadow. On the bottom line, a scene using a sphere without common boundaries. In this case, the shadow is not close to the object.

Because we do not consider rays too close to the border between the object and its shadow, only a small part of the cube shadow is used to define rays. Therefore, the resulting shape on the environment map surface has the particularity of being elongated and not correctly representing the shape of the object but only a part of it. Due to the particular place of the light in this scene, the cone is large and, even if the light source can be placed inside this area, this one is too large to define this result as a correct orientation (*i.e.*, many other light source positions are possible). The bottom line presents an ideal case where the light source is placed perfectly, and so the environment map orientation can be correctly defined.

Orientation results of the method using impostor rays' definition: Results of the method using impostor for the rays' definition are shown in figures 5.12 for the local scene and 5.13 for global scene. This method aims to replace the reference object shape into the shape defined by the intersection of the rays with the object impostor (*i.e.*, the target shape). Again, although a metric has been developed for the orientation, allowing it to be done automatically, the orientation quality of the method using impostor for the rays' definition is visual and let to the reader's discretion.

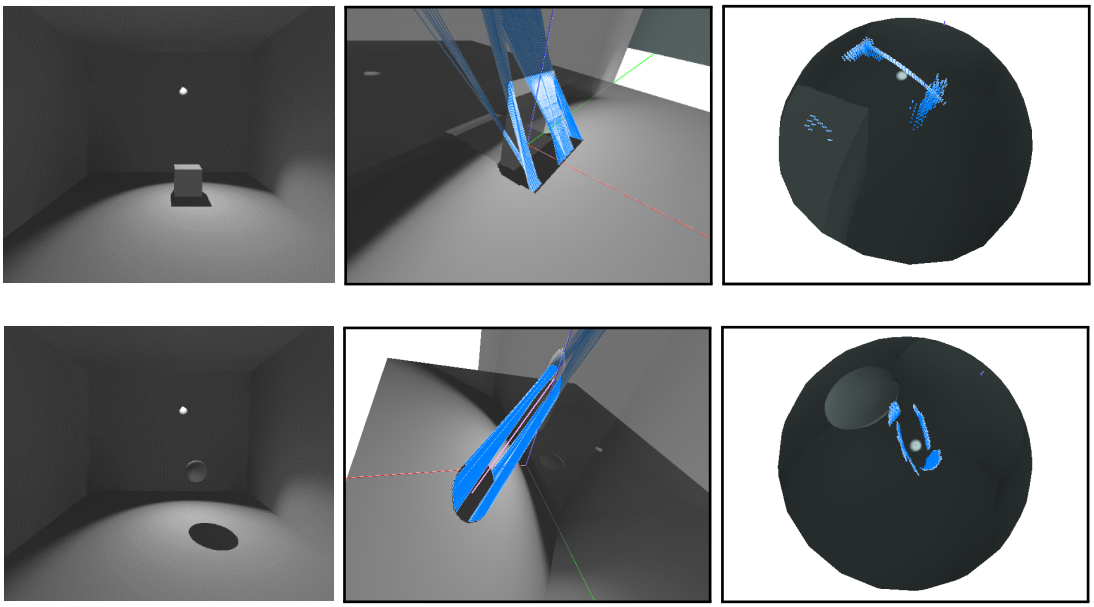


Figure 5.11 – Result of the orientation of the environment map using the 2D information for rays' definition on special cases, the light is placed behind the object. The light path used is the light path from the shadow to the reference object (see section 3.3, light path 1). The left row presents the local scene images. The middle row presents the 3D local scenes, back view. The right row presents the 3D global scene and the orientation result.

On figure 5.12, the turquoise segments show the full rays (from shadow to the light source). These have been hidden in the top left for a better understanding. The orange segments show the parts of the ray from the shadow impostor to the intersection with the object impostor. The example presented here shows the correct results, where the reference object is fully contained into the target shape and well adjusted. In this case, we can consider that the light source, the reference object, and its shadow are all aligned and that the best environment map orientation has been retrieved.

Figure 5.13 shows a local and a global scene. In this case, rays surround the object but do not match perfectly with it. Either some rays intersect the reference object shape, or a large area is visible between the object and the target shapes. However, this alignment of the light, object and shadow is preserved, and we can consider the orientation is visually correct.

Due to the variation of results for the rays definition, and due to special cases for the light position or light path definition, the orientation step is not robust enough and follows an uncertain environment map orientation. We discuss in the next section about the identified limitations for these two methods.

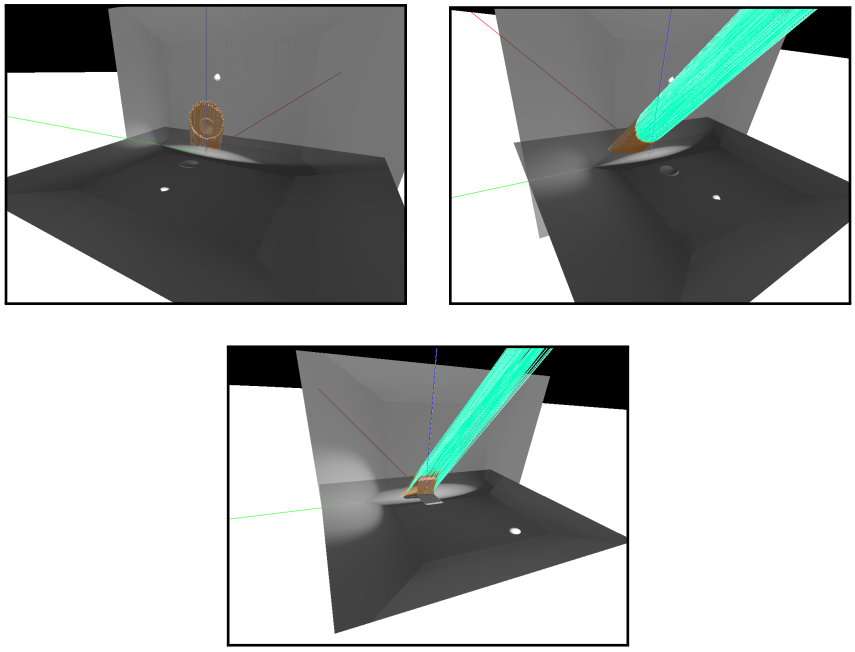


Figure 5.12 – Result of the placement of the reference object inside the area described by the set of rays, using the light path from the main light source to the shadow. Top line: final result of the placement of the reference object (a sphere) inside the area described by the rays. Lower line: same for a cube.

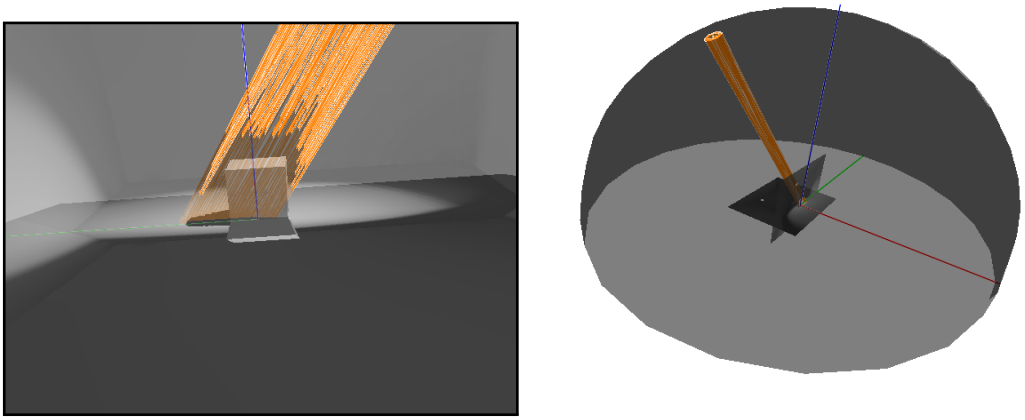


Figure 5.13 – Result of the orientation of the environment map using the light path from the light to the shadow. Left: final result of the placement of the reference object inside the area described by the rays. Right: final position of the environment map.

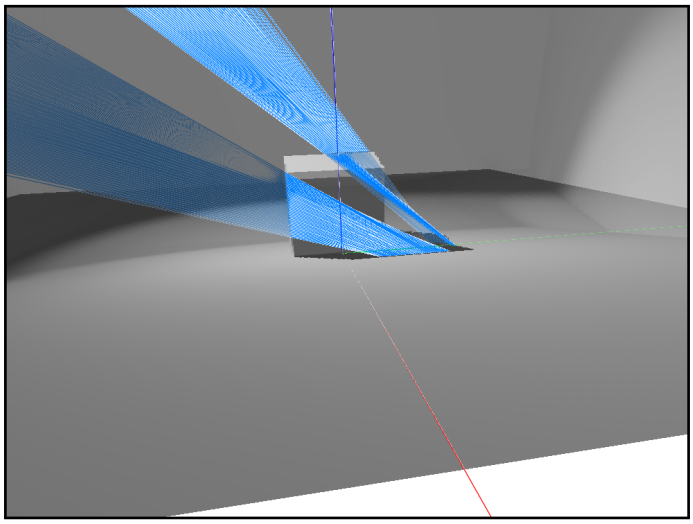


Figure 5.14 – Bad result for the 2D rays' definition using the light path from the reference object to its shadow, see section 3.3 light path 1. The method does not match correct points of the shadow with points of the object. The resulting cone is too large to define a correct and small area on the environment map.

5.5 Discussions & identified limitations

In this part, we will discuss the limitation of two methods presented here we identify. The first two methods present different problems. The primary one is linked to does not know the 3D of the object. In each case, some parts of the 3D object cannot be linked to the shadow. That is because of the camera point of view related to the object or because of the reference object real form. Indeed, the 3D object is projected in the image plan in 2D, and as a result, the depth is lost. These losses of information may be a problem during the orientation step. The figure 5.15 illustrates this problem.

5.5.1 Method using 2D rays' definition

In this section, we expose various problems linked to the 2D definition of rays, and explain the reasons why we have changed several parts to improve the results. The problems are linked to various parts of the methods. We list them in order of appearance. According to our opinion, the main problem is the last one of the list. We develop it in section 5.5.3.

- The first detected problem is that we do not have any insurance that the object point and the shadow point correspond to each other during the matching step. We have defined some criteria to avoid that (criteria 1, 2, and 3) and to ensure that the two points of a match correspond to each other and to obtain a cone of rays. However,

in several cases, the criteria are not sufficient to ensure a correct cone which contains the light source.

The figure 5.14 illustrates this problem. The method does not match the correct points of the shadow with points of the object. Indeed, top cube side points are linked to right shadow side points; and left cube side points are linked to top shadow side points. The resulting cone is too large to define a correct and small area on the environment map.

- The second problem is that the object shape does not always correspond to the shadow shape. Indeed, auto occlusion can modify the object shape in the image plane. For example, a cylinder axed in the direction of the light source projects a shadow shaped like a circle (see figure 5.15, left) but the cylinder projected in the image plane can have a rectangle shape.

  In the same way, if the cylinder is axed in the direction of the camera, the object shape is a circle, but the shadow shape (according to the light source position) can look like a rectangle (see figure 5.15, right). In these cases, it is not possible to obtain correct matches, and particularly to get a cone of rays. In the case of the scheme figure 5.15, right, the rays defined by matches converge at the object position, and it is not possible to determine the light source position.

- This is not always possible to place the light source in the cone defined by the rays due to the position of the light. Indeed, these methods present some parameters which influence the result of the set of matches (number, angle for the cone, precision of the result). Even if the value of these parameters has been defined to provide satisfactory results in most cases, results are not robust enough, particularly in special cases.

  For instance, failure cases are cases with a light source placed behind the camera. If the light, object and camera are aligned. If the object does not touch the floor (and so the shadow), but the projection on the camera plan (due to the camera or light position) presents a common border between the shadow and the object. All these cases need to adapt the parameters of the criteria, but since we want an automatic method without user help, such adaptation cannot be considered.

- The last problem we detected is, for us, the main problem of this method. We consider that the object is on a plane and not a 3D object (see figure 5.1) to recover the 3D matches. The recovered 3D matches are an approximation but not a complete 3D recovery. The figure 5.10 presents a recovered calculated area using a spherical object. The resulting area on the surface of the environment map presents some

discontinuities. These problems result in the approximation of the object to a surface. In this example, the recovering area corresponds to the shape of the object. But in some cases (especially if only a part of the object presents some matches) the recovered area has not the same shape than the object.

Even if the calculated area allows us, in most cases, to replace the light source and to find an environment map orientation, we cannot be sure that it is the correct orientation. The method using impostors for the rays' definition has been developed in a way to resolve it, but an only partial 3D method is not sufficient.

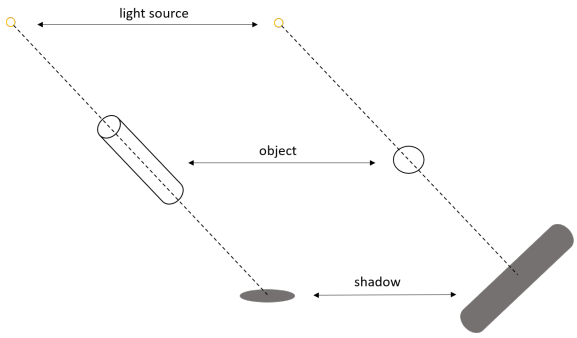


Figure 5.15 – Study case of a cylinder: problem to find some matches due to the shape difference between the object and its shadow.

5.5.2 Method using impostor for rays' definition

The changes present in the method using impostor for the rays' definition are supposed to prevent problems encountered in the full 2D method. To avoid the majorities of issues detected with the method using 2D information, we have decided to change the rays definition. This new definition, a direct 3D definition using impostors, allows avoiding all the first steps and so the problems linked to it. Indeed, if the matches are defined directly in 3D, all the issues related to the 3D recovery and previous steps (*e.g.*, the cone definition, criteria, 3D recovery) are avoided.

- Using impostors to simulate the 3D and define rays does not allow us to avoid all ray definition problems we could meet using a full 2D method. Indeed, even if the rays definition does not depend on the shape of the object because using a light path from the light to the shadow (see section 3.3), light path 2, the resulting set of rays is not insured to correctly represent the physical light path. The problem is just pushed back further, at the placement of the object in the area defined by the rays step (see section 5.3.3). To summarise, it is not ensured that the target shape, corresponding to the 2D object shape.

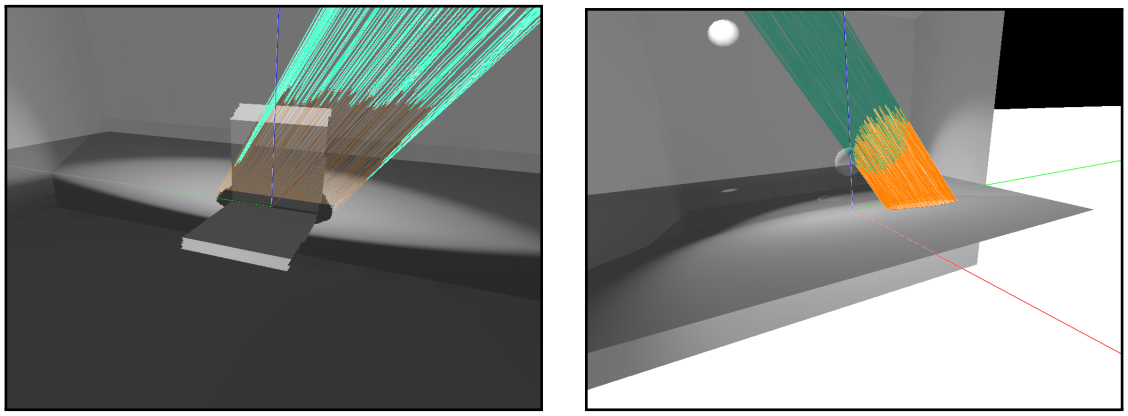


Figure 5.16 – Bad results of the placement of the reference object inside the area described by the set of rays, using the light path from the main light source to the shadow, see section 3.3, light path 2. The method fails to replace the object.

- In some cases, the shadow is placed in front of and behind the object regarding the camera position. From this, result that the shadow shape (*i.e.*, horizontal impostor), is on both sides of the object (*i.e.*, vertical impostor). Indeed, because we place the impostor intersection at the level of the basis of the object, a part of the shadow can be higher this level on the image plane, and the other one lower this level.

  In this case, some rays do not intersect the object impostor regardless of the light source position. In half of the possible light source positions (*i.e.*, 180° possible environment map orientation), a part of the rays is in this case. In the other possible light source positions, the other part of rays is in this case.

  An easy way to avoid this problem is to ignore these rays. However, in this case, the problem is to know which rays we have to ignore. Obtain such divergence (the two sets are in opposite directions) can bring us to obtain a 180° orientation error.

  In addition, obtaining two sets of rays in the opposite direction is not physically correct. Indeed, because all rays come from the same light source and define the same shadow, all rays must be parallel, and thus, in the same direction.

- If we consider all points of the shadow boundaries to define rays, some of them may be close to the object, especially if the object and shadow present a common border. In this case, those rays always intersect the object impostor. That is because the light is not placed at infinity and so, rays are not parallel. That is not physically correct. Moreover, because we do not take into account depth, rays starting from the shadow too close to the object will always intersect the depth object part which is projected on the plane.

  To avoid that, rays must be defined at a certain distance from the object. This

distance is hard to define because regarding to the object and of the real light source position.

- As for the method which defines rays using only 2D information, we consider that the object is on a plane and not a full 3D object (see figure 5.16) to recover the 3D matches. Even if we can consider the shadow shape on a plane (the ground) and so the rays set as a correct representation of the cone defined by the light, the rays set intersect a 2D impostor where the totality of the 3D object has been projected. We do not take into account the depth of the object, and the projection of a 3D object on a 2D plane modifies the shape of it. Thus, the comparison of the target shape and the object one is not physically correct.
  Again, we consider this problem as the main one even if the method using impostor for the rays' definition corrects problems linked to the previous one (*e.g.*, the rays definition).

5.5.3 Conclusion & highlighting on the importance of the real 3D of the local scene

In this chapter, we have presented two methods for environment map orientation, using full 2D, and a mixed 2D & 3D method using impostors. Those methods present correct results in simple cases, but because of a too large variation in results in more complex cases and special cases (*e.g.*, light placed at the zenith of the object), those methods are not trustworthy and cannot be generalised. Moreover, the large number of parameters, especially for the method using 2D information (*e.g.*, criteria, see section 5.2.1), but also for the method using impostor for the rays' definition (*e.g.*, the necessary distance of the object impostor to take into account rays), do not allow us to obtain good results in a large variety of cases. Those parameters must be defined case by case, depending on the scene. Those parameters are not easy to define, a small variation of these involves a large variety of results.
Finally, the too numerous limitations (presented in the previous section 5.5) of those methods lead us to think that both methods are not robust enough and do not allow to generalise them to real cases. This is why we do not develop any metric to test the accuracy of those methods.

We conclude that those methods can be seen as test methods and give us valuable information: we need the real 3D of a part of the scene or at least an object to obtain usable information to orient the environment map around the local scene. Due to the lack

of precision of 2D information to orient the environment map in a 3D space, it is important to get closer from a physical description (section 3.3, light path 3), using the 3D of the reference object and following the physical path of the light to improve the results as in chapter 4.

Chapter 6

Conclusion & future work

Résumé

Dans cette thèse, nous avons présenté la première méthode pour recaler automatiquement une carte d'environnement avec une scène locale. Dans la littérature, de nombreuses méthodes ont besoin d'orienter la carte d'environnement pour obtenir une cohérence avec une scène locale. Cette orientation est dans la majorité des cas effectuée de manière manuelle par un utilisateur. Nous présentons une méthode pour recaler automatiquement une carte d'environnement avec une scène locale. Une caractéristique de notre approche est que nous n'avons pas besoin de créer un modèle 3D complet de la scène locale, ni d'interaction avec l'utilisateur. De plus, nous ne faisons que de simples hypothèses. Notre méthode fournit des résultats corrects même dans les cas où l'objet cache une partie de l'ombre.

Nous proposons un pipeline pour créer une représentation virtuelle de la scène en utilisant nos données d'entrée. Cette représentation comprend la scène globale représentée par la carte d'environnement et la scène locale représentée par un objet de référence et son ombre. La représentation de la carte d'environnement et de l'ombre de l'objet de référence sont extraites des images d'entrée. L'objet de référence est un simple modèle 3D donné par l'utilisateur. En utilisant la connaissance de la position de la source lumineuse principale sur la carte d'environnement, nous pouvons simuler l'éclairage et projeter une ombre (dite ombre calculée) sur le sol. Il est alors possible de comparer la forme de l'ombre calculée avec celle de l'ombre d'entrée pour récupérer la position correcte de la source lumineuse principale. Cette position est définie comme celle présentant le moins de différences entre les deux ombres. L'orientation finale de la carte d'environnement est directement liée à la position de cette source lumineuse principale.

Nous fournissons une évaluation de l'approche proposée en calculant deux métriques qui comparent notre estimation d'angle avec les directions réelles au sol. Notre estimation d'orientation montre que notre méthode récupère une orientation de carte d'environnement correcte. Nous montrons aussi que nos résultats sont plus précis quand l'ombre d'entrée est complète et non partiellement cachée par l'object (*i.e.*, placé derrière). En effet, dans ce cas, la superposition des formes d'ombre ne peut pas être parfaite et de fait, la comparaison non plus.

Travaux futurs concernant le recalage automatique de cartes d'environnement : Dans nos travaux futurs, nous voulons améliorer la précision de l'orientation des cartes d'environnement. Premièrement, l'optimisation de la recherche d'orientation par l'utilisation d'algorithmes plus efficaces comme une descente de gradient améliorerait le temps

de calcul tout en augmentant la précision de l'orientation. Deuxièmement, notre approche pourrait être étendue à un plus grand nombre de cas, comme la prise en compte d'un plus grand nombre de sources lumineuses, de sources lumineuses surfaciques ainsi que la prise en compte de différents cas d'orientation de cartes d'environnement pour permettre des orientations décentrées.

Méthode d'optimisation : Actuellement, lors de l'étape de minimisation de métrique, chaque orientation possible de la carte d'environnement, ainsi que de nombreuses échelles possible sont testées pour trouver la bonne orientation. Mais, puisqu'un minimum global de la métrique utilisée (voir section 4.5) peut être trouvé (voir figure 4.6), nous pouvons supposer qu'une méthode plus rapide peut être utilisée. En effet, une méthode d'optimisation pour trouver la meilleure orientation utilisant une descente du gradient pourrait être mise en place. L'utilisation d'une telle méthode pourrait améliorer le temps de calcul en ne testant qu'un sous-ensemble des orientations possibles et converger vers le minimum global sans tester chaque possibilité.

Prise en compte de plusieurs sources lumineuses : Actuellement, notre méthode prend en compte une source lumineuse unique, la plus importante. Nous associons à cette source de lumière une ombre d'un objet de référence unique. Mais considérer le profil de lumière au complet (c'est-à-dire la totalité de la carte d'environnement) pourrait permettre d'obtenir de meilleurs résultats. De plus, prendre en compte plusieurs sources de lumière permet de considérer plusieurs ombres. Dans ce cas, le calcul de la métrique ne présenterait pas un seul minimum global, mais plusieurs minima locaux (pour tous les cas où seule une partie des sources lumineuses est alignée avec l'objet de référence et son ombre) et un unique minimum global (le cas où toutes les sources lumineuses sont alignées avec une ombre, celle qu'elles ont engendrée). Un cas qui inclurait la méthode d'optimisation et plusieurs sources lumineuses devrait alors tenir compte de ces minima locaux. Un autre avantage de prendre en compte l'ensemble du profil lumineux est que nous n'avons pas à déterminer quelle partie de la carte d'environnement est considérée comme la source lumineuse principale et quelle ombre est liée à celle-ci.

On remarque que le fait de prendre l'ensemble du profil de lumière n'exempte pas les cas particuliers. Par exemple, un cas avec deux sources lumineuses identiques diamétralement opposées sur la carte de l'environnement ne permet pas de déterminer quelle orientation est la bonne entre deux solutions possibles.

Tenir compte des sources lumineuses surfaciques : Une autre amélioration possible de notre méthode est la prise en compte des sources lumineuses surfaciques. En effet, dans la méthode actuelle, nous considérons la source lumineuse comme réduite à un point. Chaque rayon est projeté à partir de ce point et la forme de l'ombre qui en résulte en dépend. Mais sur la carte d'environnement (et dans le monde réel), la source lumineuse est représentée par une zone. Les rayons projetés par chaque point de la source lumineuse changent la forme de l'ombre qui en résulte. Considérer une telle forme plutôt qu'un point unique peut davantage correspondre à la réalité.
Nos premiers tests montrent que les surfaces sont trop grandes (voir figure 4.8) pour obtenir des formes d'ombres similaires à celles présentes dans l'image d'entrée, et donc de meilleurs résultats. Cela est dû à la position 3D de la source lumineuse dans notre représentation de la scène. En effet, la source lumineuse est plus proche que dans la scène réelle.

Travaux futurs - Approche utilisant le Machine learning : Pour compléter la méthode d'orientation présentée aux chapitres 3, 5 et 4, une méthode utilisant le Machine learning a été amorcée.

Introduction / Contexte / Problématique : L'apprentissage automatique (ou Machine learning) est un sous domaine de l'intelligence artificielle. Son but est d'approcher des fonctions arbitraires, non connues, à partir d'un grand nombre d'exemples donnés. Les modèles construits sont paramétriques et sont fixés pendant une phase dite d'entraînement du modèle. Par la suite, le modèle peut être utilisé sur de nouvelles données issues de la même distribution, qui n'ont pas été utilisées lors de l'apprentissage, pour obtenir une prédiction.
Un aspect intéressant des méthodes d'apprentissage automatique est la généralisation du modèle. Celui-ci apprend les paramètres du modèle et n'a pas uniquement pour but de mémoriser les données d'entrainement. Le modèle peut alors assimiler les besoins et les utiliser lors de la phase de prédiction, sans que ceux-ci ne soient spécifiquement énoncés.
Un second aspect intéressant est que de tels algorithmes sont conçus pour être indépendants de leur domaine d'application. Ainsi, les structures des algorithmes ayant fait leurs preuves peuvent servir pour des problèmes différents mais présentant des caractéristiques similaires.

Nous supposons que l'utilisation d'un tel algorithme peut nous permettre d'apprendre quelle orientation une carte d'environnement doit présenter pour correspondre à l'éclairage de la scène locale. Le but est ainsi de pouvoir apprendre l'éclairage de scène locale en

relation avec l'éclairage de la carte d'environnement pour pouvoir par la suite récupérer l'éclairage d'une carte d'environnement donnée et retrouver la bonne qui correspond à une scène locale donnée. Il est alors nécessaire de comprendre quels sont les paramètres importants d'une scène pour être en mesure de les apprendre.

Pour ce faire, nous devons déterminer les paramètres de lumière importants, créer le réseau de neurones qui sera capable d'apprendre ces paramètres et créer la base de données pour la phase d'apprentissage. Nous avons déterminé que la géométrie de la scène et la position des sources lumineuses sont les paramètres les plus importants. Mais, la diversité des environnements, la couleur des objets, les autres paramètres des sources de lumière (comme la taille ou la couleur) ou la position de la caméra sont aussi des paramètres importants. Nous devons en tenir compte lors de la création de la base de données.

Une telle méthode présente l'avantage de trouver rapidement une orientation de carte d'environnement correcte en fonction de l'éclairage local de la scène. De plus, si la base de données est correctement créée, et avec un nombre suffisant de cas, nous sommes en mesure de prendre en compte différents cas comme la présence de plusieurs sources de lumière ou l'absence d'objet de référence présent dans la scène locale.

Ainsi, pour obtenir des résultats corrects, il est important de définir précisément les différents paramètres qui influencent la détection de l'orientation de la carte d'environnement et de proposer plusieurs exemples qui font varier ces paramètres, pour les prendre en compte lors de l'apprentissage. Ainsi, une grande base de données de milliers d'exemples est nécessaire.

Données d'entrée : Les données d'entrée de la méthode seront les mêmes que celles décrites précédemment, voir section 3.2. En résumé, une carte d'environnement et une scène locale. Nous n'avons pas à fournir le masque d'un objet de référence ou la géométrie de celui-ci. Mais, une étape d'apprentissage doit être faite et un grand nombre d'exemples doivent être donnés. Un ensemble de couples de cartes d'environnement et de scènes locales correspondantes, présentant une orientation correcte, doit être appris. Pour obtenir les données d'entrée pour l'étape d'apprentissage, nous avons décidé de créer un ensemble de données virtuelles. En raison du grand nombre d'images nécessaires, nous avons décidé de créer un ensemble de données virtuelles utilisables pour apprendre les informations utiles. Parce que l'ensemble de données est virtuel, nous maîtrisons les paramètres de chaque scène. Cette information peut aussi être utilisée pour apprendre une estimation correcte de l'orientation de la carte d'environnement.

Création du jeu de données : Dans cette section, nous décrivons comment nous avons créé un ensemble de données pour entraîner le réseau de neurones. Nous décrivons les caractéristiques identifiées que le jeu de données doit avoir pour permettre l'orientation de la carte d'environnement. Pour créer l'ensemble du jeu de données, nous utilisons le logiciel PovRay[1]. PovRay est un outil utilisé pour créer des contenus graphiques 3D. Il permet de décrire une scène et d'en faire le rendu. Un tel outil est utile dans ce cas car il nous permet de contrôler tous les paramètres de la scène. Ainsi, la position de la lumière, les objets présents dans la scène, leurs couleurs, *etc.* peuvent être choisis, ce qui nous permet de les modifier et d'obtenir un grand nombre de cartes d'environnement qui présentent des paramètres contrôlés et connus.
Aucun jeu de données réel (ou virtuel) de ce type n'est déjà disponible et le temps de création d'un tel jeu de données est important. Pour ces deux raisons, nous avons préféré utiliser un logiciel pour créer le jeu de données.

Les paramètres que nous modifions sont le nombre de scènes virtuelles différentes que nous créons, le nombre d'images locales différentes que nous capturons pour une scène virtuelle créée, le nombre de profils lumineux et le nombre de rotations de la carte d'environnement que nous voulons considérer. Chaque scène virtuelle est créée à partir de plusieurs objets (maisons, arbres, personnages). Pour chaque scène virtuelle, le nombre de chaque type d'objet est sélectionné aléatoirement dans la scène. Chaque objet est placé dans la scène et se voit attribuer une couleur au hasard. Le sol est plat et peut présenter une légère inclinaison. Sa couleur varie entre le vert et le brun pour être plus réaliste. Le profil de ciel est choisi parmi un ensemble de profils de ciel (ensoleillé, nuageux, pluvieux et soleil-couchant), et différents nuages pour chaque scène.

La figure 6.1 présente un sous-ensemble d'un jeu de données généré avec PovRay. Les images sont par paires : une image de scène globale et une image de scène locale. Sur une ligne, toutes les images présentent le même profil d'objet (c'est-à-dire que les mêmes objets sont présents dans la scène), mais le profil lumineux et des couleurs d'objets changent. Toutes les deux lignes, le profil d'objet change (*i.e.*, nous obtenons une nouvelle scène). La figure 6.2 montre un zoom sur deux couples différents du jeu de données. Chaque ligne présente un couple. Les cartes de l'environnement se trouvent dans la colonne de gauche et les scènes locales dans la colonne de droite. On peut noter le changement des paramètres entre les deux scènes : la position de la lumière, la proximité des objets, les couleurs, etc. On peut aussi noter la variabilité de la scène locale. La premier présentant uniquement la

[1] PovRay website

façade d'un bâtiment, et la seconde des arbres et les animaux.

Ainsi, nous créons un ensemble de données de couple d'images (carte d'environnement et scène locale) qui présentent une grande variabilité. Les scènes sont cohérentes et présentent différents profils lumineux, même si les positions des objets sont aléatoires.

Approche basée Machine learning : Deux différents types de méthodes basées apprentissage peuvent être pensées. Soit le but est de trouver une carte d'environnement correcte parmi une base de données de cartes d'environnement déjà existante, soit le but est de créer une image d'une carte d'environnement possible avec un profil lumineux correspondant à la scène locale. Les deux cas retournent une image de carte d'environnement étant une bonne solution pour élairer la scène locale d'entrée. Cependant, ces méthodes sont différentes pour la conception du réseau de neurones.

Dans le premier cas, nous avons défini qu'un *CNN* est la forme correcte de réseau de neurones à adopter. Dans le second cas, un *auto-encoder* est une solution appropriée. Dans les deux cas, la littérature présente des réseaux de neurones entre cinq et sept couches pour obtenir des résultats corrects [ZL17] [MYM+17] [GSY+17]. Dans le cas d'une reconstruction d'image utilisant un auto-encoder, le même nombre de couches est utilisé pour reconstruire l'image finale et obtenir le résultat.
Nous pouvons supposer que les résultats de cette amélioration possible de notre méthode peuvent être proches des résultats présentés dans [WPL18] ou dans [HGSH+17].

Pour valider une telle approche, nous pouvons comparer nos résultats avec ceux exposés dans les travaux de Hold-Geoffroy *et al.* [HGSH+17]. Ils présentent une technique basée sur un CNN pour estimer l'éclairage extérieur à partir d'une seule image. Pour former le CNN, les auteurs ont utilisé des paires d'images ainsi que les paramètres d'éclairage (position du soleil, turbidité et exposition) du jeu de données SUN360 [XEOT12]. Enfin, sont extraites des images à faible champ de vue des panoramas pour obtenir un ensemble de données d'essai.
Ils peuvent alors détecter, à l'aide d'une seule image à faible champ de vue, les paramètres du ciel, et ainsi prédire les paramètres d'éclairage extérieur pour recréer des images à 360°. Enfin, ils utilisent la carte d'environnement ainsi obtenue, pour ré-éclairer des scènes et sont capables d'insérer de manière cohérente un objet dans l'image.

6.1 Conlusions

In this thesis, we presented the first method to register an environment map with a local scene automatically. In the literature, many methods need to orient the environment map to obtain coherence with a near scene. This orientation is done manually by a user. We present a method to register an environment map with a local scene automatically. A characteristic of our approach is that we do not need to create a full 3D model of the scene or any user interaction, and we make a few assumptions. Our method provides correct results even in cases where the object hides part of the shadow.

We propose a pipeline to create a virtual representation of the scene using a few input data. This representation includes the global scene represented by the environment map and the local scene represented by a reference object and its shadow. The representation of the environment map and the reference object shadow are extracted from input images. The reference object it obtains by a simple 3D model given by the user.
Using the knowledge of the main light source position on the environment map, we can simulate the illumination and project a computed shadow on the ground. It is possible to compare the computed shadow shape with the input one to recover the correct position of the main light source. The correct position is the one that presents fewer differences between the two shadows. The final environment map orientation (*i.e.*, registration) is directly linked to the position of this main light source.

We provide an evaluation of the proposed approach by computing two metrics which compare our angle estimation with ground truth directions. Our orientation estimation shows that our method recovers a correct environment map orientation. We also show that our results are more accurate when the input shadow is complete and not hidden by the reference object (*i.e.*, placed behind). Indeed, in this case, the shadow shapes superposition cannot be perfect neither the comparison.

6.2 General conclusion of the developed methods

We present here a conclusion for chapters 3, 5 and 4. In the literature, several methods use environment maps to proceed to one or various steps. But, to be efficient and to provide the most of their scope, these must be oriented. Indeed, whether in virtual reality, 3D reconstruction or virtual tour, for example, the environment map must be oriented to bring respectively, correct illumination, reconstruction, or places for objects in the scene.

After a study of the state of the art, we conclude that the orientation step is most of the time unnoticed, ignored, or handmade. To solve this problem, we present here an automatic environment map orientation method. Because such a process aims to be included in method pipeline, we wanted this method using a few input data.
We present our input data and generalities in section 3. Because the idea of the method is to follow the light path from the main light source in the environment map to an object present in the local scene, to recover the orientation using shadow position, we present the different ways to browse the light path we experiment here.
Moreover, we present here, the different assumption we make on input data. Some assumptions are directly linked to the input data itself, as the necessity of the presence of the main light source in the environment or the fact that we consider this environment as placed at infinity. Other assumptions are linked to our method, as the necessities to have a flat ground around the object of the local scene. That ensures the shadow we use to be undistorted by the ground geometry.

The next section (section 4) presents the method using the 3D of a reference object of the local scene. To be physically realist, this method follows the light path: from the light source to the shadow.
This method compares an input shadow with a computed one to recover the correct position of the light (and so the environment map orientation) by making coincide the two shadows.

The third section (section 5) presents methods we developed to reduce assumptions: using only 2D information, using impostor to define rays which acts as first tests and uses fewer possible input data: an environment map and a local scene image. The idea is to recover the light source direction in the local scene image using a backward light path: from the shadow to the light source. Thus, the methods aim to recover, either a light source position into a rays shape on the environment map surface or to include an object into a cone of rays.
Because of the too many possible cases and to the lack of information in the 2D image to obtain a correct 3D position for the light source (and so a correct environment map orientation), these methods are not robust and present a too low accuracy.
We conclude that the 3D information of, at least, an object in the local scene is needed to recover a correct orientation.

The table 6.1 resumes the differences between the different orientation methods presented in the previous sections. We provide the dimension number of each method, which light

Section number	Dimension number	Light Path	Input data				Main problem	Additional information
			EM	local	depth	3D model		
5.2.1.1	2D	path 1	✓	✓			to many special cases	approximate 3D rays recovery
5.2.2	2,5D	all (see 3.3)	✓	✓			to many special cases	use of two impostors
4.2.1.1	3D	path 3	✓	✓	✓		a part of the 3D is not reconstructed	need additional depth as input
4.2.1.2	3D	path 3	✓	✓		✓ (only once)	some self shadow occlusion occur	a unique reference object model is used

Table 6.1 – Summary table of the different presented methods. The different light path methods are discuss in the 3.3 section. See section for more information.

path is used, the necessary input data, the main problem and useful information. The two first lines show the preliminaries methods, using the fewer input data but with significant main problems. The two last lines show the final approaches. We divide it into two lines to present the two possibilities to obtain the necessary 3D information: the depth information, and the 3D model of an object. The last option is the one which provides better results.

To conclude, we present here a fully automatic method to register an environment map with a local scene. The method allows, with a small number of input data and few assumptions to recover a correct orientation of the environment map placed around the local scene.

6.3 Future work - Environment map orientation method

In future work, we want to improve orientation accuracy. At First, optimising the orientation research by using more efficient algorithms as gradient descent should improve the computation time while increasing orientation accuracy. Second, as discuss in 4.7, our approach should be extended to more cases, such as taking into account more light sources, area light sources, and more orientation cases of environment maps to enable out-centered environment map orientations. We present here other ones further future:

Optimisation method: Currently, each possible environment map orientation and different scales are tested to refind the correct one. But, because a global minimum of the used metric (see section 4.5) can be found (see figure 4.6), we can argue that a faster method can be used. Indeed, an optimisation method to find the best orientation as Gradient descent could be set up.
Using such a method could improve the computation time by testing only a subset of possible orientation and converge towards the global minimum without testing each possibility.

Orientation improvement: A second improvement linked to the orientation (*i.e.*, registration) step 5.3 is to first look for the orientation, and then look at the right scale, only for the better orientation. Indeed, all our tests have shown the same radius value present the minimum score for every angle. In other words, for each possible radius, the minimum metric value is at the same *theta* angle value. In this case, search first the better radius, and then, only for this radius, the better angle, improve the research time significantly. Even if the computation time is not a key point, improve this aspect is a plus for the method, mostly if this environment map orientation method is integrated into a pipeline.

6.4 Future work - Machine learning approach

To complete the orientation method presented in chapters 3, 5 and 4 a method using Machine learning have been started.

6.4.1 Introduction / Context / Problematic

Machine learning is a sub-domain of artificial intelligence. Its purpose is to approximate arbitrary, unknown functions from a large number of given examples. The models built are parametric and are fixed during a training phase. Subsequently, the model can be used on new data from the same distribution, which were not used during learning, to obtain a prediction.
An interesting aspect of automatic learning methods is the generalisation of the model. This one learns the parameters of the model and is not only intended to store the training data informations. The model can then assimilate the needs, and use them during the prediction phase, without them being specifically stated.
A second interesting aspect is that such algorithms are designed to be independent of their field of application. Thus, the structures of proven algorithms can be used for different problems with similar characteristics.

We assume that using a such algorithm may allow us to learn which orientation an environment map has to present to correspond to a local scene illumination. The aim is so to be able to learn the local scene illumination in relation with the environment map one to be able, later, to recover the environment map illumination and to refind the correct one which correspond to a given local scene. It is necessary to understand what are the important parameters in a scene to be able to learn them.

To do that, we must determine the important light parameters, create the neural nework who will be able to learn these paramaters and create the database for the learning phase. We had determined that the geometry of the scene and the light sources position are the most important parameters to vary. But, the diversity of environments, the object colors, other light source parameters (as the size or the color), camera position are also important parameters. We must take them into account during the database creation. This one is presented later.

Such a method presents the advantage to quickly find a corect environment map orientation according to the local scene illumination. Moreover, if the database is correctly created, and with a sufficient amount of cases, we are able to take into account more cases as several light sources or cases without a selected reference object present in the local scene.

But, to obtain correct results, it is important to define preciselly the different parameters which influence environment map orientation detection and to propose several examples that vary these parameters to take them into account during the learning step. Thus, a large database of thousands examples is needed.

6.4.2 Input data

The input data of the method must be the same than the ones described previously, see section 3.2. Basically, an environment map and a local scene. We do not have to provide the reference object mask or the geometry of it. But, a learning step has to be done and a large number of example must be given. A dataset of couple of environment map and corresponding local scene, presenting a correct orientation must be learn. To obtain the input data for the learning step, we decide to create a dataset. Because of the large number of images needed to learn information on image, we decide to create a virtual dataset usable to learn usefull information on image to recover a correct environment map orientation for a given local scene. Because the dataset is virtual, we master the parameters of each scene given to learn the neural network. This information can also be use to learn a correct estimation of the environment map orientation.

6.4.3 Dataset creation

In this section, we describe how we creates a dataset to train the neural network. We describe the identified characteristics the dataset must have to allow the environment map registration. To create the environment map dataset, we use PovRay[2] software. PovRay is

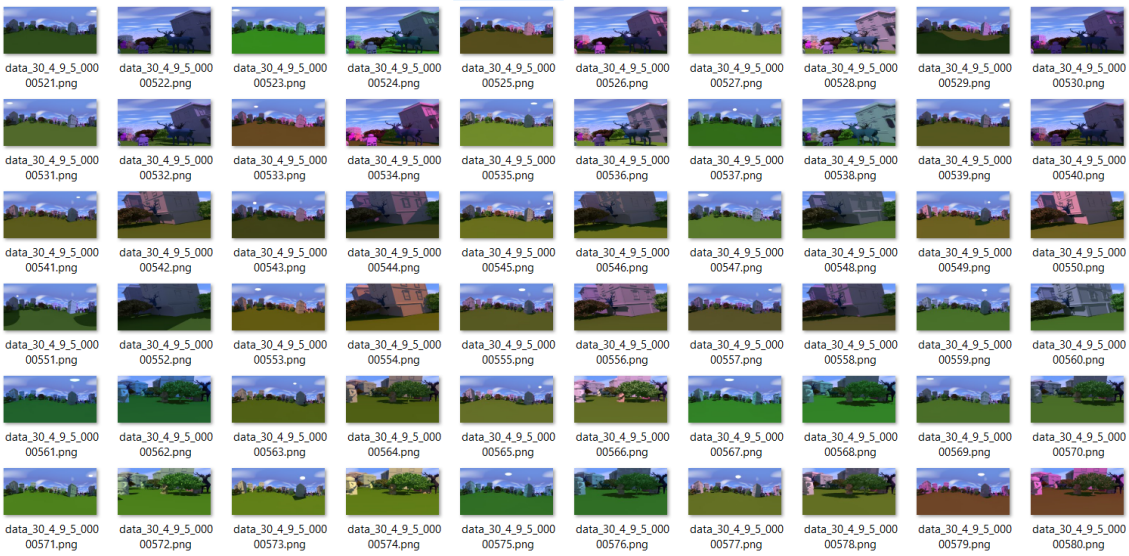


Figure 6.1 – Dataset generated with PovRay. Different scenes, light profiles, objects colors and positions, *etc.* For each scene, there is an environment map with an asociated possible local scene.

a tool used to create three-dimensional graphics contents. It allows to describe a scene and render it. A such tool is useful in this case because allowing us to control every parameter of the scene. Thus, the light position, the objects present in the scene, their color, *etc.* can be choose and permit us to change them and obtain a large number of environment map which present controlled and known parameters.
No such real (or virtual) dataset is available yet and the creation time of a such dataset is important. For these two reasons, we prefered to use a software to create the dataset.

The parameters we change are the number of different virtual scenes we create, the number of different local images we capture for a created virtual scene, the number of light profiles and the number of environment map rotations we want to consider.
Each virtual scene is created using several objects (houses, trees, characters). For each virtual scene, the number of each type of object is chosen randomly in the scene. Each object is placed randomly in the scene and the object color is also choosen randomly. The ground is flat and can present a slight inclination. Its color varies between green and brown to be more realistic. The sky profile is choosen among a set of sky profiles (sunny, cloudy, rainy and sunset) with different clouds for each scene.

The figure 6.1 presents a subset of a generated dataset using PovRay. The images work in pairs: an environment map and a local scene image. On a line, all images present the

[2] PovRay website

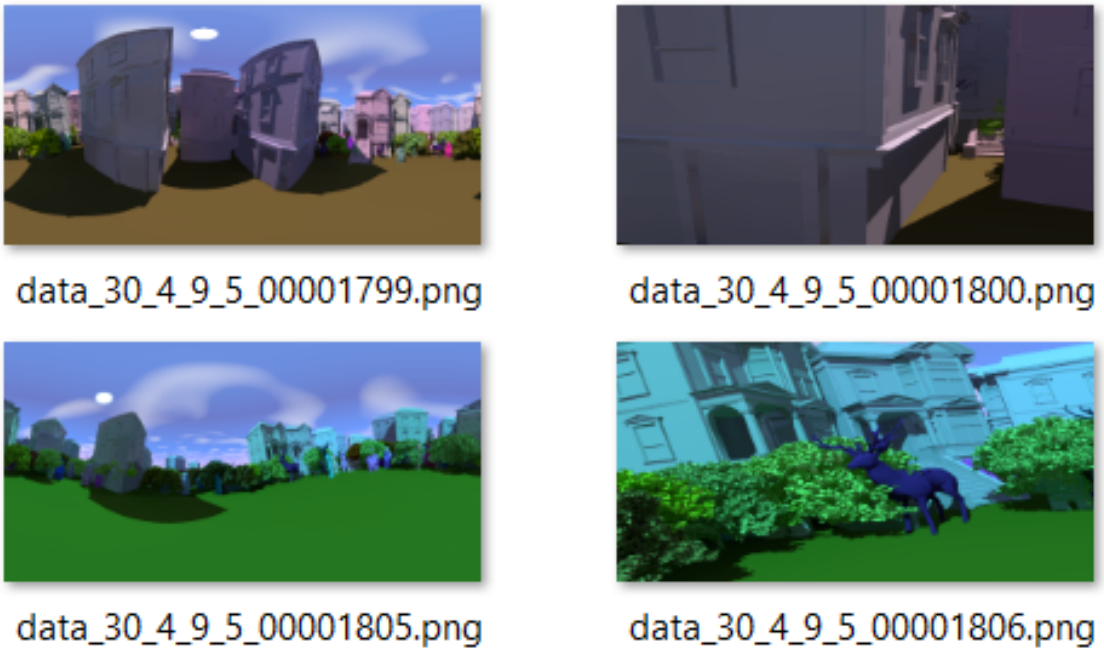


Figure 6.2 – Dataset generated with PovRay. Two different scenes (each row). Environment maps are on the left column, and local scenes on the right one.

same object profile (*i.e.*, the same objects are present in the scene) but a changing light profile and objects colors. Each two lines, the object profile changes (*i.e.*, we obtain a new scene). The figure 6.2 shows a zoom on two different couples of the dataset. Each line presents a couple. The environment maps are on the left column and the local scenes are on the right one. We can note the change of parameters between the two scenes: the light position, objects proximity, colors, *etc.* We can also note the variability of the local scenes. The first one presenting only building facade, and the second one trees and animal.

Thus, we create a dataset of image couple (environment map and local scene) which present the expected variability. The scene are coherent even if the object positions are random, with various light profiles.

6.4.4 Learning based approach

Two different types of learning-based methods can be developped. Either the goal is to find a correct environment map among a database of environment map and to select it into the database to obtain an already existing environment map, or the goal is to create an image of a possible environment map with a light profile corresponding to the local scene. Both cases return an environment map image which is a good solution to illuminate the

input local scene. However, those methods are different for the conception of the neural network.

In the first case, we define that a CNN is the correct neural network form to adopt. In the second case an auto-encoder may be an appropriate solution. In both cases, the litterature presents between five and seven layers to obtain correct results [ZL17] [MYM+17] [GSY+17]. In case of an image reconstruction using an autoencoder, the same number of layers are used to reconstruct the final image and obtain the result.

To validate such an approach, we can compare the result with those exposed in the work of Hold-Geoffroy *et al.* [HGSH+17]. They present a *"CNN-based technique to estimate high-dynamic-range outdoor illumination from a single low dynamic range image."* To train the CNN, authors had use pairs of low field-of-view images, and illumination parameters (sun position, turbidity and exposure) from the SUN360 dataset [XEOT12]. Then, they extract low field-of-view images from the panoramas to obtain a test dataset.
They can detect, using a single low field-of-view image, sky parameters, and so, to predict outdoor illumination parameters to recreate 360° images. Finaly, they use the obtained environment map to relight scenes and are able to coherently insert an object in the image.

Bibliography

[AFM+06] Amir Akbarzadeh, J-M Frahm, Philippos Mordohai, Brian Clipp, Chris Engels, David Gallup, Paul Merrell, M Phelps, S Sinha, B Talton, et al. Towards urban 3d reconstruction from video. In *Third International Symposium on 3D Data Processing, Visualization, and Transmission*, pages 1–8. IEEE, 2006.

[AKRS11] Clemens Arth, Manfred Klopschitz, Gerhard Reitmayr, and Dieter Schmalstieg. Real-time self-localization from panoramic images on mobile devices. In *10th IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 37–46. IEEE, 2011.

[ANBSE12] Nijad Al-Najdawi, Helmut E Bez, Jyoti Singhai, and Eran A Edirisinghe. A survey of cast shadow detection algorithms. *Pattern Recognition Letters*, 33(6):752–764, 2012.

[ARBJ03] Sameer Agarwal, Ravi Ramamoorthi, Serge Belongie, and Henrik Wann Jensen. Structured importance sampling of environment maps. In *ACM Transactions on Graphics (TOG)*, volume 22, pages 605–612. ACM, 2003.

[BCD+13] Francesco Banterle, Marco Callieri, Matteo Dellepiane, Massimiliano Corsini, Fabio Pellacini, and Roberto Scopigno. Envydepth: An interface for recovering local natural illumination from environment maps. In *Computer Graphics Forum*, volume 32, pages 411–420. Wiley Online Library, 2013.

[BCRA11] Adrien Bousseau, Emmanuelle Chapoulie, Ravi Ramamoorthi, and Maneesh Agrawala. Optimizing environment maps for material depiction. In *Computer graphics forum*, volume 30, pages 1171–1180. Wiley Online Library, 2011.

[BDVK12] Jean-Charles Bazin, Cédric Demonceaux, Pascal Vasseur, and Inso Kweon. Rotation estimation and vanishing point extraction by omnidirectional vision in urban environment. *The International Journal of Robotics Research*, 31(1):63–81, 2012.

[BHACB16] Houssem-Eddine Benseddik, Hicham Hadj-Abdelkader, Brahim Cherki, and Samia Bouchafa. Direct method for rotation estimation from spherical images using 3d mesh surfaces with spharm representation. *Journal of Visual Communication and Image Representation*, 40:708–720, 2016.

[BKDV08] Jean-Charles Bazin, Inso Kweon, Cedric Demonceaux, and Pascal Vasseur. A robust top-down approach for rotation estimation and vanishing points extraction by catadioptric vision in urban environment. In *IROS 2008. IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 346–353. IEEE, 2008.

[BL07] Matthew Brown and David G Lowe. Automatic panoramic image stitching using invariant features. *International journal of computer vision*, 74(1):59–73, 2007.

[BN76] James F Blinn and Martin E Newell. Texture and reflection in computer generated images. *Communications of the ACM*, 19(10):542–547, 1976.

[BTH15] Dash Bodington, Jayant Thatte, and Matthew Hu. Rendering of stereoscopic 360 views from spherical image pairs. 2015.

[BTVG06] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *European conference on computer vision*, pages 404–417. Springer, 2006.

[CCC08] Massimiliano Corsini, Marco Callieri, and Paolo Cignoni. Stereo light probe. In *Computer Graphics Forum*, volume 27, pages 291–300. Wiley Online Library, 2008.

[CJAMJ05] Petrik Clarberg, Wojciech Jarosz, Tomas Akenine-Möller, and Henrik Wann Jensen. Wavelet importance sampling: Efficiently evaluating products of complex functions. *ACM Transactions on Graphics (Proceedings of SIGGRAPH)*, 24(3):1166–1175, aug 2005.

[CLG+18] Dan A Calian, Jean-François Lalonde, Paulo Gotardo, Tomas Simon, Iain Matthews, and Kenny Mitchell. From faces to outdoor light probes. In *Computer Graphics Forum*, volume 37, pages 51–61. Wiley Online Library, 2018.

[CMS87] Brian Cabral, Nelson Max, and Rebecca Springmeyer. Bidirectional reflection functions from surface bump maps. In *ACM Siggraph Computer Graphics*, volume 21, pages 273–281. ACM, 1987.

[CON99] Brian Cabral, Marc Olano, and Philip Nemec. Reflection space image based rendering. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 165–170. ACM Press/Addison-Wesley Publishing Co., 1999.

[CWW11] Matthäus G Chajdas, A Weis, and Rüdiger Westermann. Assisted environment map probe placement. In *Proceedings of SIGRAD 2011. Evaluations of Graphics and Visualization—Efficiency; Usefulness; Accessibility; Usability; November 17-18; 2011; KTH; Stockholm; Sweden*, number 065, pages 17–25. Linköping University Electronic Press, 2011.

[DCL11] Jean-Lou De Carufel and Robert Laganiere. Matching cylindrical panorama sequences using planar reprojections. In *IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 320–327. IEEE, 2011.

[DE02] Fadi Dornaika and James Elder. Image registration for foveated omnidirectional sensing. In *European Conference on Computer Vision*, pages 606–620. Springer, 2002.

[Deb98] Paul Debevec. Rendering synthetic objects into real scenes: Bridging traditional and image-based graphics with global illumination and high dynamic range photography. In *Proceedings of the 25th annual conference on Computer graphics and interactive techniques*, pages 189–198. ACM, 1998.

[Deb06] Paul Debevec. Reflection mapping history. http://www.pauldebevec.com/ReflectionMapping/, 2006. [Online; Accessed 2019-October-27].

[Deb08] Paul Debevec. Rendering synthetic objects into real scenes: Bridging traditional and image-based graphics with global illumination and high dynamic range photography. In *ACM SIGGRAPH 2008 classes*, page 32. ACM, 2008.

[DLR16] Aleksandar M Dimitrijević, Martin Lambers, and Dejan Rančić. Comparison of spherical cube map projections used in planet-sized terrain rendering. *Facta Universitatis, Series: Mathematics and Informatics*, 31(2):259–297, 2016.

[DSSD97] Katja Daubert, Hartmut Schirmacher, François X. Sillion, and George Drettakis. *Hierarchical Lighting Simulation for Outdoor Scenes*, pages 229–238. Springer Vienna, Vienna, 1997.

[Ecl] Ecliptique. Chenonceau castle panorama. http://www.ecliptique.com/chenonceau/index.html. [Online; Accessed 2019-October-27].

[FAG+] Blengini Filippo, Bacchilega Alessandra, Bethaz Guido, La Torre Matteo, and Clauss Roland. Mont Blanc panorama, in2white company. http://www.in2white.com/. [Online; Accessed 2019-October-27].

[Gaba] Acoca Gabriel. Mars panorama. https://www.gabrielacoca.fr/visite-virtuelle-mars/index.html. [Online; Accessed 2019-October-27].

[Gabb] Acoca Gabriel. Venezia panorama. https://www.gabrielacoca.fr/venezia/index.html. [Online; Accessed 2019-October-27].

[GEM07] Thorsten Grosch, Tobias Eble, and Stefan Mueller. Consistent interactive augmentation of live camera images with correct near-field illumination. In *Proceedings of the 2007 ACM symposium on Virtual reality software and technology*, pages 125–132. ACM, 2007.

[Gre86] Ned Greene. Environment mapping and other applications of world projections. *IEEE Computer Graphics and Applications*, 6(11):21–29, 1986.

[GRR+16] Stamatios Georgoulis, Konstantinos Rematas, Tobias Ritschel, Mario Fritz, Tinne Tuytelaars, and Luc Van Gool. Natural illumination from multiple materials using deep learning. *arXiv preprint arXiv:1611.09325*, 2016.

[GRR+18] Stamatios Georgoulis, Konstantinos Rematas, Tobias Ritschel, Efstratios Gavves, Mario Fritz, Luc Van Gool, and Tinne Tuytelaars. Reflectance and natural illumination from single-material specular objects using deep learning. *IEEE transactions on pattern analysis and machine intelligence*, 40(8):1932–1947, 2018.

[GSY+17] Marc-André Gardner, Kalyan Sunkavalli, Ersin Yumer, Xiaohui Shen, Emiliano Gambaretto, Christian Gagné, and Jean-François Lalonde. Learning to predict indoor illumination from a single image. *ACM Transactions on Graphics (SIGGRAPH Asia)*, 9(4), 2017.

[GTTC03] Duke Gledhill, Gui Yun Tian, Dave Taylor, and David Clarke. Panoramic imaging—a review. *Computers & Graphics*, 27(3):435–445, 2003.

[Gue] Park Guell. Carretera del carmel panorama. https://www.parkguell.es/tour/. [Online; Accessed 2019-October-27].

[HBT15] Matthew Hu, Dash Bodington, and Jayant Thatte. Ee368 project proposal 360 rendering of stereoscopic 3d views from spherical image pairs. 2015.

[HCP07] Michal Havlena, Kurt Cornelis, and Tomáš Pajdla. *Towards city modeling from omnidirectional video*. 2007.

[HGSH+17] Yannick Hold-Geoffroy, Kalyan Sunkavalli, Sunil Hadap, Emiliano Gambaretto, and Jean-François Lalonde. Deep outdoor illumination estimation. In *IEEE International Conference on Computer Vision and Pattern Recognition*, volume 2, 2017.

[HHJ+15] Jan Heller, Michal Havlena, Michal Jancosek, Akihiko Torii, and Tomas Pajdla. 3d reconstruction from photographs by cmp sfm web service. In *14th IAPR International Conference on Machine Vision Applications (MVA)*, pages 30–34. IEEE, 2015.

[HRP80] B Heinz, W Ried, and HD Pflug. The lumisphere, a new model of pre-biotic evolution. *Naturwissenschaften*, 67(4):178–181, 1980.

[HS99] Wolfgang Heidrich and Hans-Peter Seidel. Realistic, hardware-accelerated shading and lighting. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 171–178. ACM Press/Addison-Wesley Publishing Co., 1999.

[HSSL05] Justin Hensley, Thorsten Scheuermann, Montek Singh, and Anselmo Lastra. Interactive summed-area table generation for glossy environmental reflections. In *ACM SIGGRAPH 2005 Sketches*, page 34. ACM, 2005.

[HW12] Lukas Hosek and Alexander Wilkie. An analytic model for full spectral sky-dome radiance. *ACM Transactions on Graphics (TOG)*, 31(4):95, 2012.

[IM05] John R Isidoro and Jason L Mitchell. Angular extent filtering with edge fixup for seamless cubemap filtering. In *ACM SIGGRAPH 2005 Sketches*, page 33. ACM, 2005.

[Jac07] Katrien Jacobs. *Illumination for mixed reality of complex-to-model scenes*. University of London, University College London (United Kingdom), 2007.

[JCJ09] Wojciech Jarosz, Nathan A. Carr, and Henrik Wann Jensen. Importance sampling spherical harmonics. *Computer Graphics Forum (Proceedings of Eurographics)*, 28(2):577–586, apr 2009.

[Jef] Martin Jeffrey. Old Town Square panorama. http://360gigapixels.com/prague_gigapixel_panorama_900K_2018/. [Online; Accessed 2019-October-27].

[JL06] Katrien Jacobs and Céline Loscos. Classification of illumination methods for mixed reality. In *Computer Graphics Forum*, volume 25, pages 29–51. Wiley Online Library, 2006.

[JNVL10] Katrien Jacobs, Anders Hjorth Nielsen, Jeppe Vesterbaek, and Céline Loscos. Coherent radiance capture of scenes under changing illumination conditions for relighting applications. *The Visual Computer*, 26(3):171–185, 2010.

[JSZ+19] Xin Jin, Xing Sun, Xiaokun Zhang, Hongbo Sun, Ri Xu, Xinghui Zhou, Xiaodong Li, and Ruijun Liu. Sun orientation estimation from a single image using short-cuts in dcnn. *Optics & Laser Technology*, 110:191–195, 2019.

[Kán15] Peter Kán. Interactive hdr environment map capturing on mobile devices. In *Eurographics (Short Papers)*, pages 29–32, 2015.

[KDS04] Jan Kautz, Katja Daubert, and Hans-Peter Seidel. Advanced environment mapping in vr applications. *Computers & Graphics*, 28(1):99–104, 2004.

[KH13a] Hansung Kim and Adrian Hilton. 3d scene reconstruction from multiple spherical stereo pairs. *International journal of computer vision*, 104(1):94–116, 2013.

[KH13b] Hansung Kim and Adrian Hilton. Planar urban scene reconstruction from spherical images using facade alignment. In *IVMSP Workshop, 2013 IEEE 11th*, pages 1–4. IEEE, 2013.

[KH14] Hansung Kim and Adrian Hilton. Hybrid 3d feature description and matching for multi-modal data registration. In *IEEE International Conference on Image Processing (ICIP)*, pages 3493–3497. IEEE, 2014.

[KH15] Hansung Kim and Adrian Hilton. Block world reconstruction from spherical stereo image pairs. *Computer Vision and Image Understanding*, 139:104–121, 2015.

[KHFH11] Kevin Karsch, Varsha Hedau, David Forsyth, and Derek Hoiem. Rendering synthetic objects into legacy photographs. *ACM Transactions on Graphics (TOG)*, 30(6):157, 2011.

[Kin05] Gary King. Real-time computation of dynamic irradiance environment maps. *GPU Gems*, 2:167–176, 2005.

[KM00] Jan Kautz and Michael D McCool. Approximation of glossy reflection with prefiltered environment maps. In *Graphics Interface*, volume 2000, pages 119–126, 2000.

[KSF+13] M Koehl, A Schneider, E Fritsch, F Fritsch, A Rachedi, and S Guillemin. Documentation of historical building via virtual tour: the complex building of baths in strasbourg. *ISPRS-International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, XL-5 W*, 2:385–390, 2013.

[LE10] Jean-François Lalonde and Alexei A Efros. Synthesizing environment maps from a single image. Technical report, Citeseer, 2010.

[LEN10] Jean-François Lalonde, Alexei A Efros, and Srinivasa G Narasimhan. Detecting ground shadows in outdoor consumer photographs. In *European conference on computer vision*, pages 322–335. Springer, 2010.

[LFTG97] Eric P. Lafortune, Sing-Choong Foo, Kenneth E. Torrance, and Donald P. Greenberg. Non-linear approximation of reflectance functions. In *Proc. SIGGRAPH '97*, volume 31, pages 117–126, 1997.

[LK10] Robert Laganiere and Florian Kangni. Orientation and pose estimation of panoramic imagery. *Mach Graph Vis*, 19(3):339–363, 2010.

[LN15] Maxime Lhuillier and Thanh-Tin Nguyen. Synchronization and self-calibration for helmet-held consumer cameras, applications to immersive 3d modeling and 360 video. In *International Conference on 3D Vision (3DV)*, pages 434–442. IEEE, 2015.

[Low99] David G Lowe. Object recognition from local scale-invariant features. In *The proceedings of the seventh IEEE international conference on Computer vision*, volume 2, pages 1150–1157. Ieee, 1999.

[LSD+14] Jia Li, Yehua Sheng, Ping Duan, Siyang Zhang, and Haiyang Lv. Constructing 3d model based on panoramic images. In *Advances in Computer Science and its Applications*, pages 1061–1065. Springer, 2014.

[MD03] Ameesh Makadia and Kostas Daniilidis. Direct 3d-rotation estimation from spherical images via a generalized shift theorem. In *Proceedings. 2003 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages II–217. IEEE, 2003.

[MD06] Ameesh Makadia and Kostas Daniilidis. Rotation recovery from spherical images without correspondences. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(7):1170–1175, 2006.

[MdP] Paris Musées and Mairie de Paris. Museum panorama, Museosphere. http://museosphere.paris.fr/. [Online; Accessed 2019-October-27].

[MHS18] Rafael Monroy, Matis Hudon, and Aljosa Smolic. Dynamic environment mapping for augmented reality applications on mobile devices. *arXiv preprint arXiv:1809.08134*, 2018.

[MK09] Branislav Micusik and Jana Kosecka. Piecewise planar city 3d modeling from street view panoramic sequences. In *IEEE Conference on Computer Vision and Pattern Recognition, 2009. CVPR*, pages 2906–2912. IEEE, 2009.

[MLH02] David K McAllister, Anselmo Lastra, and Wolfgang Heidrich. Efficient rendering of spatial bi-directional reflectance distribution functions. In *Proceedings of the ACM SIGGRAPH/EUROGRAPHICS conference on Graphics hardware*, pages 79–88. Eurographics Association, 2002.

[MSD04] Ameesh Makadia, Lorenzo Sorgi, and Kostas Daniilidis. Rotation estimation from spherical images. In *ICPR 2004. Proceedings of the 17th International Conference on Pattern Recognition*, volume 3, pages 590–593. IEEE, 2004.

[MYM+17] David Mandl, Kwang Moo Yi, Peter Mohr, Peter M Roth, Pascal Fua, Vincent Lepetit, Dieter Schmalstieg, and Denis Kalkofen. Learning lightprobes for mixed reality illumination. In *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 82–89. IEEE, 2017.

[NN04] Ko Nishino and Shree K. Nayar. Eyes for relighting. *ACM Trans. Graph.*, 23(3):704–711, August 2004.

[NRH03] Ren Ng, Ravi Ramamoorthi, and Pat Hanrahan. All-frequency shadows using non-linear wavelet lighting approximation. In *ACM Transactions on Graphics (TOG)*, volume 22, pages 376–381. ACM, 2003.

[Pel10] Fabio Pellacini. envylight: an interface for editing natural illumination. *ACM Transactions on Graphics (TOG)*, 29(4):34, 2010.

[Pov] PovRay. Povray software. http://www.povray.org/. [Online; Accessed 2019-October-27].

[PSHZ+15] Federico Perazzi, Alexander Sorkine-Hornung, Henning Zimmer, Peter Kaufmann, Oliver Wang, S Watson, and M Gross. Panoramic video from unstructured camera arrays. In *Computer Graphics Forum*, volume 34, pages 57–68. Wiley Online Library, 2015.

[PSS99] Arcot J Preetham, Peter Shirley, and Brian Smits. A practical analytic model for daylight. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 91–100. ACM Press/Addison-Wesley Publishing Co., 1999.

[PWH] Matt Pharr, Jakob Wenzel, and Greg Humphreys. Pbrt software. https://www.pbrt.org/. [Online; Accessed 2019-October-27].

[R+09] Ravi Ramamoorthi et al. Precomputation-based rendering. *Foundations and Trends® in Computer Graphics and Vision*, 3(4):281–369, 2009.

[RH01] Ravi Ramamoorthi and Pat Hanrahan. An efficient representation for irradiance environment maps. In *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 497–500. ACM, 2001.

[RH02] Ravi Ramamoorthi and Pat Hanrahan. Frequency space environment map rendering. In *ACM Transactions on Graphics (TOG)*, volume 21, pages 517–526. ACM, 2002.

[RHD+10] Erik Reinhard, Wolfgang Heidrich, Paul Debevec, Sumanta Pattanaik, Greg Ward, and Karol Myszkowski. *High dynamic range imaging: acquisition, display, and image-based lighting*. Morgan Kaufmann, 2010.

[RZZY17] Lingyan Ran, Yanning Zhang, Qilin Zhang, and Tao Yang. Convolutional neural network-based robot navigation using uncalibrated spherical images. *Sensors*, 17(6):1341, 2017.

[SBC15] Zi Siang See, Mark Billinghurst, and Adrian David Cheok. Augmented reality using high fidelity spherical panorama with hdri. In *SIGGRAPH Asia 2015 Mobile Graphics and Interactive Applications*, page 14. ACM, 2015.

[SBRCD17] Pỳnar Satỳlmỳs, Thomas Bashford-Rogers, Alan Chalmers, and Kurt Debattista. A machine-learning-driven sky model. *IEEE computer graphics and applications*, 37(1):80–91, 2017.

[SCMS08] Davide Scaramuzza, Nicolas Criblez, Agostino Martinelli, and Roland Siegwart. Robust feature extraction and matching for omnidirectional images. In *Field and Service Robotics*, pages 71–81. Springer, 2008.

[SKS02] Peter-Pike Sloan, Jan Kautz, and John Snyder. Precomputed radiance transfer for real-time rendering in dynamic, low-frequency lighting environments. In *ACM Transactions on Graphics (TOG)*, volume 21, pages 527–536. ACM, 2002.

[SLS05] Peter-Pike Sloan, Ben Luna, and John Snyder. Local, deformable precomputed radiance transfer. In *ACM Transactions on Graphics (TOG)*, volume 24, pages 1216–1224. ACM, 2005.

[Sno10] Ben Snow. Terminators and iron men: Image-based lighting and physical shading at ilm. *part of "Physically Based Shading Models in Film and Game Production," SIGGRAPH*, 2010.

[SPY11] Tomokazu Sato, Tomas Pajdla, and Naokazu Yokoya. Epipolar geometry estimation for wide-baseline omnidirectional street view images. In *IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 56–63. IEEE, 2011.

[SS97] Richard Szeliski and Heung-Yeung Shum. Creating full view panoramic image mosaics and environment maps. In *Proceedings of the 24th annual conference on Computer graphics and interactive techniques*, pages 251–258. ACM Press/Addison-Wesley Publishing Co., 1997.

[SSI99] Imari Sato, Yoichi Sato, and Katsushi Ikeuchi. Acquiring a radiance distribution to superimpose virtual objects onto a real scene. *IEEE transactions on visualization and computer graphics*, 5(1):1–12, 1999.

[SSL10] Andres Sanin, Conrad Sanderson, and Brian C Lovell. Improved shadow removal for robust person tracking in surveillance scenarios. In *20th International Conference on Pattern Recognition (ICPR)*, pages 141–144. IEEE, 2010.

[SSL12] Andres Sanin, Conrad Sanderson, and Brian C Lovell. Shadow detection: A survey and comparative evaluation of recent methods. *Pattern recognition*, 45(4):1684–1695, 2012.

[Str] Google company Streetview. Angkor-wat panorama. https://www.google.com/maps/about/behind-the-scenes/streetview/treks/angkor/#angkor-wat. [Online; Accessed 2019-October-27].

[SY10] Tomokazu Sato and Naokazu Yokoya. Efficient hundreds-baseline stereo by counting interest points for moving omni-directional multi-camera system. *Journal of Visual Communication and Image Representation*, 21(5-6):416–426, 2010.

[Sze96] Richard Szeliski. Video mosaics for virtual environments. *IEEE computer Graphics and Applications*, 16(2):22–30, 1996.

[THP09] Akihiko Torii, Michal Havlena, and Tomáš Pajdla. From google street view to 3d city models. In *IEEE 12th international conference on Computer vision workshops (ICCV Workshops)*, pages 2188–2195. IEEE, 2009.

[THP11] Akihiko Torii, Michal Havlena, and Tomáš Pajdla. Omnidirectional image stabilization for visual object recognition. *International Journal of Computer Vision*, 91(2):157–174, 2011.

[TITO15] Hajime Taira, Yuki Inoue, Akihiko Torii, and Masatoshi Okutomi. Robust feature matching for distorted projection by spherical cameras. *Information and Media Technologies*, 10(3):478–482, 2015.

[UGY07] Jonas Unger, Stefan Gustavson, and Anders Ynnerman. Spatially varying image based lighting by light probe sequences. *The Visual Computer*, 23(7):453–465, 2007.

[Uyt17] Matt Uyttendaele. Optimizing 360 photos at scale. 2017.

[Vat] Musei Vaticani. Vatican panorama. http://www.vatican.va/various/cappelle/sistina_vr/index.html. [Online; Accessed 2019-October-27].

[Ven] Ventdautan. Bordeaux panorama. https://www.bordeaux-tourisme.com/Decouvrir-Bordeaux/Visite-virtuelle-Bordeaux-360. [Online; Accessed 2019-October-27].

[VH13] Jonathan Ventura and Tobias Höllerer. Structure and motion in urban environments using upright panoramas. *Virtual Reality*, 17(2):147–156, 2013.

[WH13] Alexander Wilkie and Lukas Hošek. Predicting sky dome appearance on earth-like extrasolar worlds. In *Proceedings of the 29th Spring Conference on Computer Graphics*, pages 145–152. ACM, 2013.

[Whi79] Turner Whitted. An improved illumination model for shaded display. In *ACM SIGGRAPH Computer Graphics*, volume 13, page 14. ACM, 1979.

[Wil83] Lance Williams. Pyramidal parametrics. In *Acm siggraph computer graphics*, volume 17, pages 1–11. ACM, 1983.

[WPL18] Henrique Weber, Donald Prévost, and Jean-François Lalonde. Learning to estimate indoor lighting from 3D objects. In *2018 International Conference on 3D Vision (3DV)*, pages 199–207. IEEE, 2018.

[WT05] Tai-Pang Wu and Chi-Keung Tang. A bayesian approach for shadow extraction from a single image. In *Tenth IEEE International Conference on Computer Vision, 2005. ICCV*, volume 1, pages 480–487. IEEE, 2005.

[WWL07] Liang Wan, Tien-Tsin Wong, and Chi-Sing Leung. Isocube: Exploiting the cubemap hardware. *IEEE Transactions on Visualization & Computer Graphics*, 4:720–731, 2007.

[XEOT12] Jianxiong Xiao, Krista A Ehinger, Aude Oliva, and Antonio Torralba. Recognizing scene viewpoint using panoramic place representation. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2695–2702. IEEE, 2012.

[ZL17] Jinsong Zhang and Jean-François Lalonde. Learning high dynamic range from outdoor panoramas. *arXiv preprint arXiv:1703.10200*, 2017.

[ZLW+18] Zhe Zhu, Jiaming Lu, Minxuan Wang, Songhai Zhang, Ralph R Martin, Hantao Liu, and Shimin Hu. A comparative study of algorithms for realtime panoramic video blending. *IEEE Transactions on Image Processing*, 27(6):2952–2965, 2018.

[ZZM17] YanXiang Zhang, ZiQiang Zhu, and PengFei Ma. Walk through a museum with binocular stereo effect and spherical panorama views. In *International Conference on Culture and Computing (Culture and Computing)*, pages 20–23. IEEE, 2017.

Titre
Orientation automatique de carte d'environnement autour d'une scène locale

Résumé

Dans cette thèse, nous présentons une méthode pour recaler automatiquement une carte d'environnement avec une scène locale. Dans la littérature, de nombreuses méthodes ont besoin d'orienter la carte d'environnement pour obtenir une cohérence avec une scène locale. Cette orientation est dans la majorité des cas effectuée de manière manuelle par un utilisateur. Une caractéristique de notre approche est que nous n'avons pas besoin de créer un modèle 3D complet de la scène locale, ni d'interagir avec l'utilisateur.Nous proposons un pipeline pour créer une représentation virtuelle de la scène en utilisant nos données d'entrée. Cette représentation comprend la scène globale représentée par la carte d'environnement et la scène locale représentée par un objet de référence et son ombre. En utilisant la connaissance de la position de la source lumineuse principale sur la carte d'environnement, nous pouvons simuler l'éclairage et projeter une ombre sur le sol. Il est alors possible de comparer la forme de l'ombre calculée avec celle de l'ombre d'entrée pour récupérer la position correcte de la source lumineuse principale. L'orientation finale de la carte d'environnement est directement liée à la position de cette source lumineuse principale. Nous fournissons une évaluation de l'approche proposée en calculant deux métriques qui comparent notre estimation d'angle avec les directions réelles de référence. Notre estimation d'orientation montre que notre méthode récupère une orientation de carte d'environnement correcte.

Mots-clés
Carte d'environnement, éclairage, caméra 360, panorama, source de lumière virtuelle, image sphérique

Title
Automatic environment map registration around a local scene

Abstract

In this thesis, we present a method to automatically register an environment map with a local scene. In the literature, many methods need to orient the environmental map to be coherent with a local scene. This orientation is mostly done manually by a user. A characteristic of our approach is that we do not need to create a complete 3D model of the local scene nor interact with the user. We propose a pipeline to create a virtual representation of the scene using our input data. This representation includes the global scene represented by the environment map and the local scene represented by a reference object and its shadow. By knowing the position of the main light source on the environment map, we can simulate the lighting and project a computed shadow on the ground. It is possible to compare the computed shadow shape with the input one to recover the correct position of the main light source. The final orientation of the environment map is directly related to the position of this main light source. We provide an evaluation of the proposed approach by calculating two metrics that compare our angle estimate with actual ground truth directions. Our orientation estimation shows that our method recovers a correct environment map orientation. In this thesis, we are interested in real input data. The environment map and the local scene are extracted from photographs or videos, which already contain a lighting rendering. It is therefore important to orient the environment map in a way that is consistent with the existing lighting in the local scene. We propose an automatic method, to orient an environment map to a local scene. This method is driven by the behavior of light, drawing rays of light towards an object and attempting to match two shadows, one given as input and one calculated. We also use 3D data from the object we are considering. The originality is that we base our method on how light behaves in order to calculate and match shadows. By matching the shadows, we can estimate the correct position of the environment map.

Keywords
Environment map, relighting, illumination, camera 360, panorama, virtual light sources, spherical image

Discipline
Informatique

Unité de Recherche
Unité de recherche EA 3804 CENTRE DE RECHERCHE EN STIC (CRESTIC)
UFR SCIENCES EXACTES ET NATURELLES
Moulin de la Housse
51687 REIMS CEDEX 2