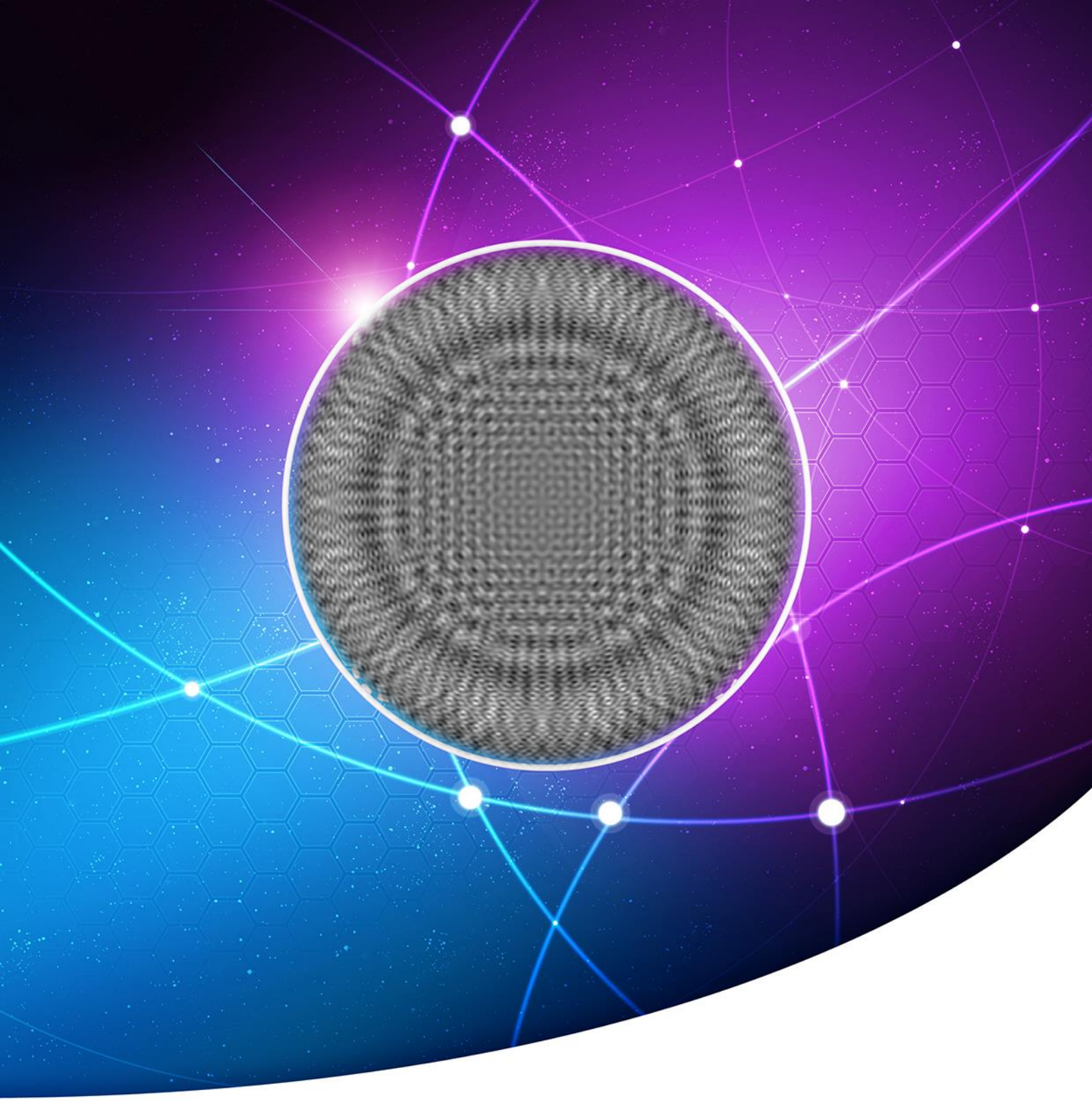


Holographie Numérique

Générer des hologrammes d'objets modélisés par ordinateur.

Wissem HAMIDOU Paul FLEUTRY Benjamin LAZARD

TELECOM ParisTech

Encadré par **Renaud GABET**

Table des matières

Introduction :

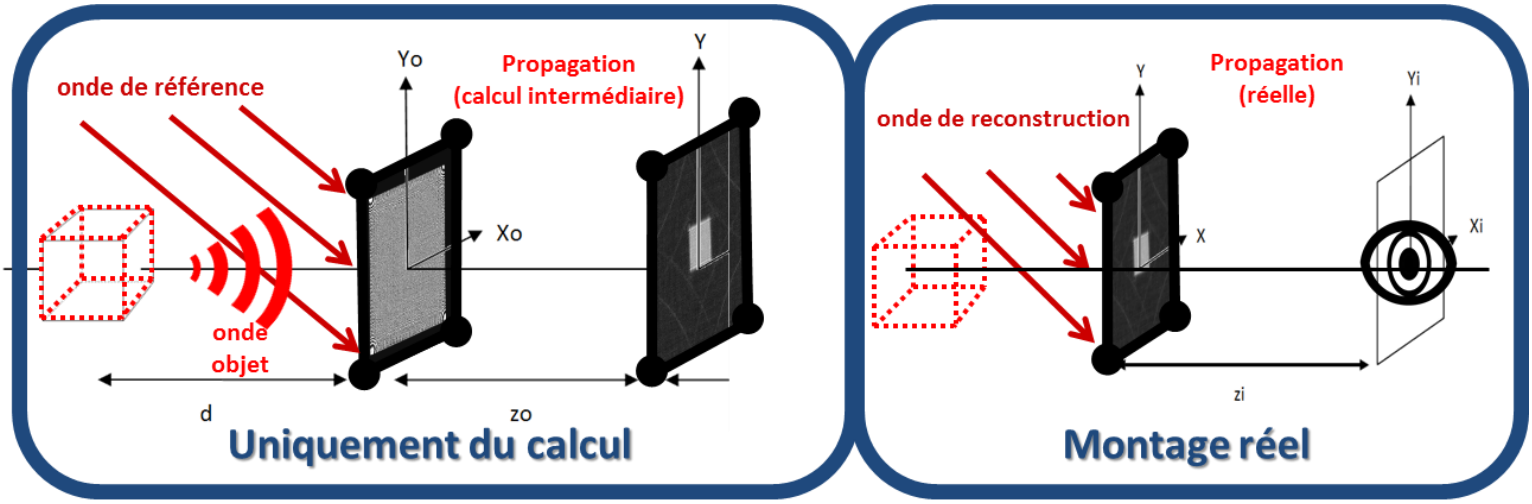
Au cours de ce projet, réalisé dans le cadre du cours **PAF** de Télécom Paristech, nous avons travaillé 2 semaines à temps plein sous l'égide de Renaud GABET. Le but était de reprendre le projet commencé il y a trois ans et amélioré chaque année par des étudiants de l'école, en y apportant des modifications suggérées par plusieurs publications universitaires (voir la bibliographie).

L'**holographie** est un procédé de photographie en 3D basé sur l'enregistrement de la phase et de l'amplitude de l'onde diffractée par un objet, lorsqu'éclairé par une lumière cohérente issue d'un laser.

L'**holographie numérique** est une technique d'imagerie optique qui permet de générer numériquement des hologrammes pour obtenir une vue tridimensionnelle d'un objet (modélisé en 3D) en temps réel. On utilise pour cela une matrice LCD afin d'afficher l'hologramme calculé. Ci-dessous, mieux qu'un long discours, quelques images pour illustrer les espoirs suscités par cette technologie.

A l'issue du projet, nous avons réalisé plusieurs fonctions Matlab qui permettent successivement de générer quelques objets en 3D en tant que nuage de points, puis d'en calculer l'hologramme en 2 étapes. Nous utilisons en fait un plan intermédiaire sur lequel nous appliquons stricto sensu les équations correspondant à l'enregistrement analogique d'un hologramme, puis nous faisons un calcul de propagation de cette image jusqu'au plan de l'hologramme. L'intérêt de cette technique est de gagner du temps de calcul.

1. Montage et notations associées



Sur le schéma ci-dessus, on peut voir les différentes étapes de la construction de l'hologramme. De gauche à droite :

a. L'objet 3D

On commence par modéliser un objet comme un nuage de points. Tous ces points constituent un échantillonnage de l'objet dont on veut réaliser l'hologramme. On le conçoit dans le programme comme une sélection de certains points dans une matrice cubique de taille N_m^3 (pixels^3), ou encore L_m^3 (m^3).

b. Le WRP

Wavefront Recording Plane : il s'agit d'un plan intermédiaire entre l'objet 3D et l'écran LCD. Son intérêt est purement calculatoire, et n'a aucun intérêt physique. Chaque point de ce plan a une intensité proportionnelle à la somme des amplitudes complexes de tous les points du cube (qui sont considérés comme des sources lumineuses d'ondes planes), en interférence avec une onde de référence. Le plan du WRP fait N_0^2 (pixels²), ou encore L_0^2 (m²). Toutefois l'image « utile » en son centre fait N_w^2(pixels²), ou encore L_w^2 (m²).

c. L'écran LCD

Liquid Cristal Display, aussi appelé SLM : ou encore CGH : Computer Generated Hologram. Il s'agit de la « plaque holographique numérique ». La particularité de cet écran LCD, c'est son pas : chaque pixel mesure 8 µm de côté, ce qui permet de faire diffracter la lumière d'un laser dessus. Cet écran est configurable par un ordinateur relié en HDMI, et le but du projet, c'est de calculer l'image à afficher sur cet écran pour obtenir dans le plan (ou plutôt la zone) de reconstruction le meilleur aperçu possible de l'objet modélisé en 3D.

Cet élément intervient donc à la fois dans le calcul, puisqu'on calcule l'image à afficher, et dans le montage réel, puisqu'on fait diffracter le laser sur cette image (les niveaux de noir et blanc sont en fait des niveaux d'opacité binaires pour la lumière du laser). Il mesure N^2 (pixels²), soit L^2 (m²).

Concrètement, l'image du SLM est calculée par propagation de Fresnel de l'image du plan WRP, à l'aide de Fast Fourier Transform (cf 3.).

d. La zone de reconstruction

C'est là qu'on place l'œil, derrière le SLM sur lequel la lumière du laser diffracte pour obtenir un aperçu de l'objet 3D. Ce plan mesure Li² (m²).

e. Les distances entre les plans

Voir le schéma précèdent.

Dans 3., on explique comment calculer les différents paramètres (distances et nombre de points d'échantillonnage) de manière optimale.

2. Programmes Matlab® expliqués

Notre programme consiste essentiellement en la fonction holocubeV15, qui fait elle-même appel à d'autres fonctions que nous avons créées. Elles sont énumérées ci-dessous dans l'ordre d'appel.

a. HolocubeV15*('shape', rho)*

Entrées

Ce programme prend une forme (shape) en entrée et un paramètre ajustable rho ≥ 1 (qui sert à modifier les distances caractéristiques du montage). En pratique nous avons toujours utilisé ρ=1.

Les formes possibles sont :

1. pt : un point au centre du SLM.
2. 2pts : deux points dans le plan du SLM.
3. ptdd : deux points l'un derrière l'autre (selon l'axe optique $\vec{z}$).
4. carrevide : un carré à proprement parler.
5. carre : un carré plein.
6. cercle.
7. tube : un cylindre vu de face.
8. cube : un cube vue de face.
9. sphere.
10. Une image de nom "name" dans le dossier "lettres" en tapant 'imgname' pour shape.
    Exemple : Pour l'image A.png, exécuter ***holocubeV15('imgA', 1);***
11. Une fonction **Z=f(X,Y)** qui sera tracée en 3 dimensions et observable sur le montage. Les opérations * et ^ devront être notées .* et .^.
    Exemple : ***holocubeV15('X.^2 +Y.^2',1);*** devrait donner une demi-sphère de rayon 1.

Sorties

Du fait de ses appels aux autres fonctions décrites plus bas, elle renvoie les images correspondant à :

1. Une représentation graphique de l'objet 3D.
2. Le WRP dans le fichier outWRP.png dans le même dossier que le programme.

3. Le SLM dans le fichier outFresnel.png dans le même dossier que le programme.

b. shape3D('shape', N_m, padding, pas_pixel)

Ce programme sert à simuler des objets en 3D sous la forme d'un nuage de points.

Entrées

1. Le même argument 'shape' que précédemment.
2. N_m, le côté en pixels de la zone contenant l'objet 3D.
3. padding, la "marge" en pixels entre les bords de la zone et l'objet.
4. pas_pixel, la taille d'un pixel en mètres sur le SLM.

Sorties

Les coordonnées de chacun de ces points sont ensuite enregistrées dans une liste object = $\begin{bmatrix} x_1 & y_1 & z_1 \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot \\ x_n & y_n & z_n \end{bmatrix}$, où n est le nombre de points de l'objet, et où les valeurs d'une ligne sont les coordonnées d'un point de l'objet en m. L'origine du repère est dans le coin "en bas à gauche" (nous l'avons modifié plusieurs fois) du cube de taille L_m^3 défini en 1.a. Le nuage de points correspondant est ensuite tracé sur la figure 1.
Note : parfois, il faut triturer le zoom et la rotation pour voir l'objet ainsi formé.

c. ob2wrp(M, N, Lw, Lo, d, k)

Ce programme simule le principe de Huygens-Fresnel sur l'objet 3D : on considère que chaque point de l'objet émet une onde sphérique de la longueur d'onde avec laquelle on l'a éclairé et ainsi, on enregistre la figure d'interférence obtenue dans le plan du WRP.

Entrées

1. M est l'object du programme shape3D.
2. N, la longueur totale souhaitée en pixels du côté du WRP, incluant le padding.
3. Lw, longueur en mètres de la figure d'interférence sur le WRP, sans le padding.
4. Lo, longueur totale en mètres du WRP, avec le padding.
5. d, distance en mètres entre l'objet et le WRP.
6. k, vecteur d'onde.

Sorties

WRP : l'image correspondant au plan du WRP.

d. SFFT(Img, Lo, lambda, zo)

Ce programme prend en argument la figure d'interférence du WRP et simule sa propagation sur la distance z0 à l'aide de la méthode de la SFFT (Propagation de Fresnel, cf. 3).

Entrées

1. Img, l'image de la figure de diffraction obtenue avec ob2wrp.
2. Lo, la taille désirée de Img en mètres (indépendamment du nombre de pixels).
3. lambda, la longueur d'onde utilisée dans tout le montage (ici celle du laser).
4. zo, distance de propagation en mètres.

Sorties

WRP : l'image correspondant au plan du WRP.

Nous avons également codé de petits programmes utiles, mais qui ne méritent pas d'être détaillés (soit parce qu'ils sont triviaux, soit parce qu'ils ne sont pas absolument nécessaires).

cl*oserp2(x)* : trouve la première puissance de 2 supérieure à x.

Reconst_DFFt(Img,L, Lo, lambda, zo, Gy,p) *:* Permet de reconstruire numériquement l'image à partir de la figure de diffraction obtenue. (Cela permet de voir si l'image est bonne à partir de la figure de diffraction mais il vaut mieux tester directement sur le montage).

3. Eléments de calcul (physique et informatique)

a. Note pour la compréhension scientifique

i. Propagation de Fresnel

La propagation est en fait une diffraction, et on peut utiliser les sources secondaires de Huygens pour modéliser cela. La méthode utilisée ici est cependant différente, et se nomme "spectre angulaire".

En fait, pp.33-39 du livre en Anglais, à partir des équations de Maxwell, on trouve une relation dans le domaine fréquentiel entre le plan de l'image d'origine, et le plan de l'image après propagation sur une distance **d**, aboutissant à **l'**équation de Kirchhoff and Rayleigh-Sommerfeld ([2.48] du livre en anglais).

L'approximation de Fresnel consiste à considérer **d** grand, et les angles petits (par rapport à l'axe optique). On peut alors faire plusieurs développements limités.

Ainsi, pour une onde sphérique : $U(\vec{r}) = \frac{A}{\|\vec{r}\|} e^{j\vec{k}.\vec{r}-\omega_k t} \simeq \frac{A}{R} e^{jf(\vec{k}.\vec{r})-\omega_k t}$, où R est une constante et f est une une fonction qui représente un développement asymptotique, la composante en $\vec{z}$ de $\vec{r}$ étant proche de R.

En combinant l'approximation de Fresnel avec l'équation [2.48], on obtient 2 algorithmes pour calculer la propagation sur une distance d, selon que l'on voit l'intégrale double obtenue comme une convolution ou une TF. Si on note U(x,y) l'amplitude complexe au point de coordonnées x,y du plan du SLM, et U_o(x,y) l'amplitude complexe du point de coordonnées x0,y0 du plan du WRP,

- la méthode SFFT (Simple Fast Fourier Transform) donne :

$$Uo(xo, yo) = \frac{\exp(\mathrm{jkd})}{j\lambda d} \exp\left[\frac{jk}{2d}(x^2 + y^2)\right]$$

$$\times \iint_{-\infty}^{+\infty} \{ U_o(x_o, y_o) \exp\left[\frac{jk}{2d}({x_o}^2 + y_o^2)\right] \} \exp[-\frac{jk}{d}(x_o x + y_o y)]\, dx_o dy_o$$

- la méthode DFFT (Double Fast Fourier Transform) donne :

$$U(x,y) = \frac{\exp(\mathrm{jkd})}{j\lambda d} \iint_{-\infty}^{+\infty} U_o\,(x_o, y_o) \exp\left\{\frac{jk}{2d}\left[(x-x_o)^2 + (y-y_o)^2\right]\right\} dx_o dy_o$$

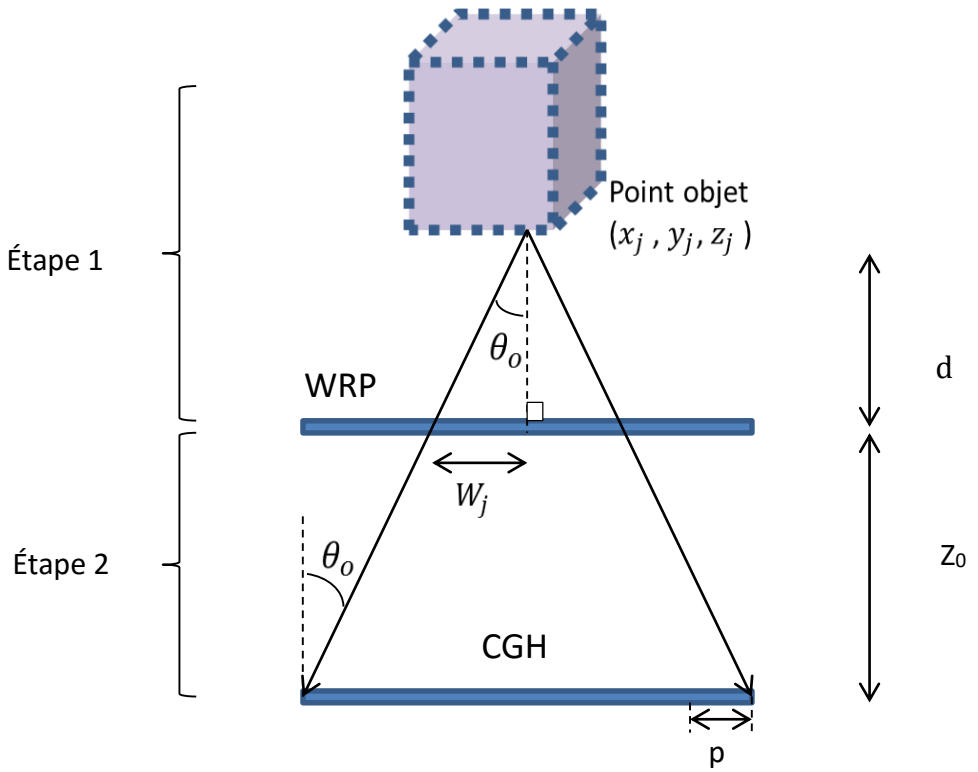
Soit : $U(x,y) = \frac{\exp(\mathrm{jkd})}{j\lambda d} F^{-1}\{F\{U_o\,(x_o, y_o)\}.F\{h(x,y)\}\}$

Avec $\mathrm{h(x,y)} = \exp\left[\frac{jk}{2d}(x^2+y^2)\right]$, ce qui permet de faire intervenir la TF2D, (algorithme fft2 dans Matlab® qui est très rapide).

On remarque toutefois que l'approximation de Fresnel impose que le laser qui joue le rôle de l'onde de reconstruction ait un faible angle d'attaque (< 3° environ), et impose également une certaine distance entre les 2 plans.

Le choix SFFT /DFFT peut se faire simplement en considérant z_0. En effet, la SFFT est une opération du domaine spatial, qui requiert, pour que le théorème de Shannon soit respecté, que $z_0 \geq \frac{L_0^2}{k\lambda}$ (pp.84-86 du livre en Anglais). La DFFT, elle, est une opération dans le domaine fréquentiel (où les distances spatiales sont inversées), et requiert pour que le théorème de Shannon soit respecté, que $z_0 \leq \frac{L_0^2}{k\lambda}$ (pp.84-86 du livre en Anglais).

ii. Principe du WRP



Le WRP est calculé selon le principe des sources de Huygens : chaque point du cube (ou de l'objet 3D en général) échantillonné est en réalité une source de lumière, et chaque

point du WRP contient l'information sur la somme des amplitudes complexes de tous les points du cube. Soit :

$u_w(x_w, y_w) = \sum_j^N \frac{A_j}{R_{wj}} \exp\left(i \frac{2\pi}{\lambda} R_{wj}\right) + onde\ de\ référence\ (x_w, y_w)$, où Rwj = $\|\overrightarrow{p_j p_w}\|$ avec $p_j(x_j,y_j,z_j)$, un point de l'objet et $p_w(x_w,y_w)$, un point du plan WRP.

Toutefois, en considérant que Rwj est proche de d, on peut se placer dans le cadre de l'approximation de Fresnel, et écrire

$R_{wj} = \sqrt{(x_w - x_j)^2 + (y_w - y_j)^2 + {d_j}^2} \approx d_j + \frac{(x_w - x_j)^2 + (y_w - y_j)^2}{2d_j}$ dans l'exponentielle, et $R_{wj} \approx d_j$ dans l'amplitude.

On pourrait utiliser ce calcul directement depuis l'objet sur le CGH, et on obtiendrait alors un hologramme donnant un résultat très correct. L'ennui, c'est que ce calcul est extrêmement chronophage. Pour l'accélérer, nous plaçons le WRP entre l'objet et le CGH, et nous appliquons simplement la SFFT ou DFFT (selon la distance z_0, pour respecter le théorème d'échantillonnage, cf. 3ai) au WRP pour obtenir le CGH.

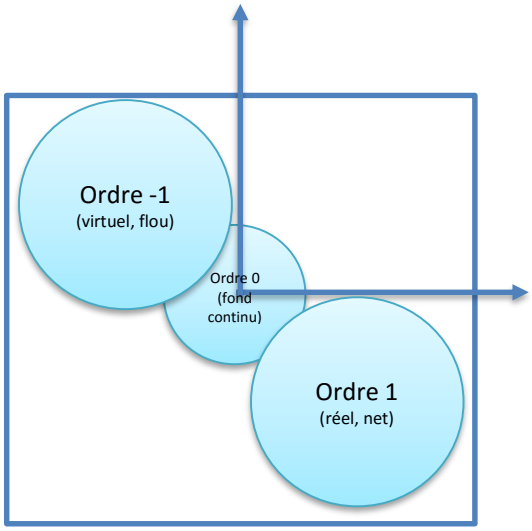
L'intérêt réside donc dans le fait que la taille utile du WRP en nombre de pixels (N_w^2) est plus petite que la taille du CGH (c'est le triangle de Thalès qu'on aperçoit dans le schéma ci-dessus). On utilise la formule suivante pour trouver L_w à L fixé (L est connu, c'est la largeur de la matrice LCD) : sin (θ_o)= 2p avec p = $\frac{L}{N}$, pas d'échantillonnage de l'image construite sur le SLM.

On trouve alors $\frac{N\lambda}{L} = \frac{L_w}{\sqrt{(d^2 + \frac{L_w^2}{4})}} \Leftrightarrow L_w = \frac{d\lambda}{\sqrt{\left(\frac{L}{N}\right)^2 - \frac{\lambda^2}{4}}}$.

iii. Choix des distances et du nombre de points d'échantillonnage.

On obtient à l'aide de ces formules une relation entre les distances du schéma en **1**. qui permet d'avoir une image nette dans la zone d'observation (équation [5.9] du livre en Anglais) :

La figure observée dans la zone d'observation est a priori la suivante :



Pour que l'ordre 1 soit net, nous avons besoin que z_i vérifie l'équation suivante :

$z_i = -(\frac{1}{z_o} + \frac{1}{z_c} - \frac{1}{z_r})^{-1}$ où z_c et z_r sont respectivement les distances au centre du CGH des sources de l'onde de reconstruction et de l'onde de référence. Si ces ondes sont considérées comme étant planes, on a $z_i \simeq -z_0$

Par ailleurs, afin d'éviter les recouvrements entre les ordres qui empêcheraient de voir correctement, et de garantir que l'image reformée soit de taille respectable, on utilise les formules suivantes ([5.19], [5.20] du livre en anglais) :

$$z_o = \frac{4\rho LLo}{\lambda N} \text{ et } L_i = \frac{\lambda G_y z_o N}{L} = 4\rho G_y Lo \, .$$

Dans ces équations, $\rho \geq 1$ est un paramètre expérimental, et G_y est le grandissement (qui vaut environ 1 si les ondes de référence et de reconstruction sont considérées comme étant planes, et que $z_i = z_0$).

A présent, on s'intéresse au nombre de points minimal qu'on doit calculer dans le WRP. L'équation [3.16] du livre en anglais fournit : $L_o = L = \sqrt{\lambda d N}$.

Etant donné que les dimensions du CGH sont fixes et connues, de même que le nombre de pixels du CGH, les équations vues jusqu'ici permettent de déterminer à peu près toutes les grandeurs du problème. La seule latitude de choix réside dans la forme de l'hologramme et le paramètre ρ , d'où les paramètres de **holocubeV15**. Le paramètre d a été cherché expérimentalement, et considéré comme optimal pour $d = \frac{z_0}{5}$, ce qui a donc été codé directement dans le programme.

iv. Méthode d'élimination de l'ordre 0

Lorsque, nous avons réalisé l'hologramme, nous avons réalisé qu'il y avait une grosse tache lumineuse derrière l'image reconstruite. Nous avons pensé qu'il pouvait s'agir de l'ordre 0, et avons donc tâché de le retirer.

En fait, si l'onde objet est $A_o = a_o(x,y)e^{i\varphi_o(x,y)}$ où $a_o(x,y)$ est l'amplitude de l'onde objet, et $\varphi_o(x,y)$, sa phase, et que l'onde de référence est $A_r = a_r(x,y)e^{i\varphi_r(x,y)}$.

L'intensité I(x, y) sur le SLM s'exprime :

$$I(x,y) \propto |A_o(x,y) + A_r(x,y)|^2 = \underbrace{|a_o|^2 + |a_r|^2}_{\text{Ordre 0}} + a_o a_r{}^* e^{-i(\varphi_r - \varphi_o)} + a_r a_o{}^* e^{i(\varphi_r - \varphi_o)} \, .$$

Sur le montage, nous avons rajouté un filtrage passe-haut pour s'assurer de la suppression du fond continu.

b. Note pour la compréhension du programme

i. Enregistrement des images

Nous ne sommes pas encore familiers avec les méthodes d'enregistrement d'images de Matlab®, nous avons par exemple remarqué de meilleurs résultats en prenant pour le SLM la valeur absolue de l'image (où chaque point est une amplitude complexe), plutôt qu'en en prenant la partie réelle. Mais pour le WRP, il semble que la partie réelle donne de meilleurs résultats. Il faut en tout cas utiliser imagesc, colormap(grey) qui donne un résultat en niveaux de gris, et surtout pas imwrite, où chaque pixel ne peut prendre que 2 $valeurs \in \{noir, blanc\}$.

ii. Meshgrid

Meshgrid est une méthode qui permet de générer des matrices pour évaluer une fonction à 2 variables sur un ensemble de valeurs. [X,Y]=meshgrid(1:1:N,1:1:N), permet de récupérer les matrices X et Y pour la suite du calcul. Pour représenter z=f(x,y), une fonction à 2 variables, il suffit de faire Z=f(X,Y), en adaptant les opérations scalaires aux matrices ("*" devient ".*", etc.) pour récupérer Z, une matrice de taille NxN, telle que Z(i,j)=f(i,j). En fait X est une matrice dont toutes les lignes sont identiques à la première (dans l'exemple, matrice de taille N^2 dont la $1^{ère}$ ligne est 1:1:N), et Y est une matrice dont toutes les colonnes sont identiques à la première (dans l'exemple, matrice de taille N^2 dont la $1^{ère}$ colonne est 1:1:N).

L'intérêt de meshgrid, c'est d'évaluer rapidement Z sur un grand nombre de valeurs, plutôt que de faire une double boucle for. En plus Matlab optimise vraiment les calculs matriciels, et on gagne donc du temps.

i. Accélération matérielle

Pour accélérer les calculs, les documents que nous avons lus suggéraient de faire appel à la GPU (la carte graphique ou Graphical Processing Unit) et au multithreading (pour exploiter les multiples cœurs du processeur). Il se trouve qu'en Matlab®, le multithreading est réalisé automatiquement, et que l'usage de la GPU s'est révélé trop compliqué à gérer lorsque nous nous sommes penché dessus (celle-ci, n'accepte que des calculs assez simples, et nous n'avons pas eu le temps de tout convertir).

L'une des publications suggérait d'utiliser une LUT (Look Up Table), qui est une table de données en mémoire où sont stockées les valeurs d'une fonction quelconque pour certaines valeurs. Le but est de diminuer le temps de calcul, en accédant à la plus proche valeur de la LUT plutôt que de recalculer la fonction fréquemment. Nous avons tellement de variables cependant, qu'à moins de faire une LUT de dimensions imposantes (10 par exemple), nous ne parviendrons à aucune amélioration. Nous avons finalement abandonné cette idée, mais elle gagnerait peut-être à être exploitée l'année suivante.

4. Expériences et Résultats

a. Test des différentes formes

Sans le WRP, pour obtenir les figures de diffraction idéales nous avons placé le plan WRP à la distance z0 du SLM. On récupère ainsi dans le plan du SLM l'aperçu de l'hologramme, et dans le plan du WRP la figure qu'on voudrait obtenir avec la méthode normale (cf. dossier correspondant).

b. Valeur optimale de z_0

Pour la SFFT et pour la DFFT nous avons fait varier z0. C'est bien la valeur calculée qui donne le meilleur résultat.

c. Valeur optimale de d

A z0 fixé par l'équation vue précédemment (ce qui n'est pas forcément une bonne idée), nous avons obtenu des résultats corrects avec d=z0/5 environ.

d. Suppression de l'ordre 0

Avec différents angles d'incidence de l'onde de référence, nous avons essayé de supprimer le fond continu. Il n'y a pas eu d'effet manifeste. Il faut vérifier sans le filtre réel sur l'axe optique et comparer pour voir s'il y a bien eu effet.

5. Conclusion :

Cette année, nous avons implémenté une méthode (l'utilisation du WRP), qui permet de diminuer le temps de calcul, nous avons diversifié les figures possibles, et nous avons découpé le programme en sous-fonctions pour bien séparer les différentes étapes de calcul. Malheureusement, par rapport à l'année précédente, nos hologrammes sont un peu moins clairs, et surtout, l'effet de profondeur n'est pas très visible.

Conseils pour l'année prochaine : il faudrait supprimer les 3 boucles for dans la fonction ob2wrp et essayer de faire un calcul matriciel plus rapide à la place. Nous avons également travaillé dans shape3D la possibilité de faire tourner les objets autour des axes selon un angle donné, mais il faut l'adapter (en recentrant l'origine après rotation, et en mettant tous les points sur des coordonnées cohérentes avec le pas_pixel). Il faut faire très attention à la manière dont sont enregistrées les images (niveaux de gris, phase ou amplitude, etc.). Nous pensons que ceci nuit beaucoup à la qualité de nos résultats. Il faut vraiment choisir la méthode optimale.

Pour innover, on pourrait essayer de faire un découpage en éléments finis plutôt qu'un nuage de points pour réaliser l'hologramme en un temps de calcul moindre.

Bibliographie :

- *Cours Optique et Photonique COM101 1ère année*, par Renaud Gabet, Télécom ParisTech (2015)
- *Rapid calculation algorithm of Fresnel CGH using LUT and WRP methods for 3D display*, par Tomoyoshi Shimobaba, Hirotaka Nakayama, Nobuyuki Masuda, et Tomoyoshi Ito, Chiba University (2010)
- Thèse : *Computer Generated Holography,* par James B. Wendt, Pomona College (2009)
- *Holographie numérique : Principe, algorithmes et applications*, par Pascal Picart et Jun-Chang Li (2012)
- *Digital Holography,* par Pascal Picart et Jun-Chang Li (2013)
- *Solution to the Twin Image Problem in Holography*, par Tatiana Latychevskaia et Hans-Werner Fink, University of Zurich (2007)
- *Fast Computation of Fresnel Holograms Employing Difference*, par Hiroshi Yoshikawa, Nihon University (2001)
- *Methods of Digital Holography: A Comparison*, par Thomas M. Kreis, Mike Adams et Werner P. O. Jüptner, BIAS (1997)
- *Digital Holographic Capture and Optoelectronic Reconstruction for 3D Displays*, par Damien P. Kelly, David S. Monaghan, Nitesh Pandey, Tomasz Kozacki, Aneta Michalkiewicz, Grzegorz Finke, Bryan M. Hennelly et Malgorzata Kujawinska, National University of Ireland et Warsaw Univsersity of Technology (2009)