



**POLYTECHNIQUE
MONTRÉAL**

UNIVERSITÉ
D'INGÉNIERIE

PHS2223 – INTRODUCTION À L'OPTIQUE MODERNE

Équipe : 04

Expérience 1

Microscopie confocale

Présenté à

Guillaume Sheehy

Esmat Zamani

Par :

Émile **Guertin-Picard** (2208363)

Laura-Li **Gilbert** (???????)

Tom **Dessauvages** (???????)

24 septembre 2024

Département de Génie Physique
Polytechnique Montréal

Table des matières

1	Introduction	1
2	Théorie	1
2.1	Microscopie Confocale	1
3	Méthodologie	1
4	Hypothèses	1
5	Annexes	3
5.1	Modèle mathématique pour le calcul de résolution axiale	3
5.2	Code source	5

1 Introduction

2 Théorie

2.1 Microscopie Confocale

La microscopie confocale est une technique d'imagerie avancée qui permet l'obtention d'images à hautes résolution, en éliminant la lumière hors-focus, particulièrement ceux à l'extérieur du plan focal. Le fonctionnement de cette technologie repose sur les principes d'illumination et de détection limités à un certain volume par la présence d'un simple trou, nommé sténopé (Source 1). En d'autres termes, une source de lumière, telle qu'un laser, est émise à travers le système, celle-ci est, ensuite, dirigée vers des miroirs dichroïques qui permettent la sélection de certaines longueurs d'onde réfléchies (Source 2). Une fois redirigée, la lumière réfléchie par les miroirs passe à travers une première lentille, concentrant ainsi le faisceau en un point précis, soit l'objet. Une fois arrivée à l'échantillon, une partie de ce faisceau est réfléchi par l'objet et, par le fait même, retransmis vers les miroirs. La lumière provenant de l'échantillon est alors redirigée vers le sténopé, situé derrière une deuxième lentille qui permet la convergence du rayon lumineux. Celle-ci traverse, ensuite, le sténopé, petite ouverture permettant d'éliminer les faisceaux ne provenant pas du plan focal, et atteint le détecteur. Puisque l'oeil nu n'est pas en mesure de traiter l'image résultante, le détecteur reconstruit indirectement l'image à l'aide de balayages de l'échantillon. De cette manière, le faisceau se déplace, selon les différentes positions axiales, sur l'objet afin d'obtenir une image complète (Source 3).

3 Méthodologie

4 Hypothèses

Afin de prédire le comportement de la résolution axiale en fonction des différents paramètres physiques du système, un modèle de calcul ainsi qu'un programme python a été utilisé. Ces derniers sont présentés en annexe. Le programme python utilise d'abord la librairie numpy afin de définir des fonctions qui génèrent des matrices de translation ou de lentille mince. Ensuite, une fonction est créée pour calculer la résolution axiale en fonction de la largeur du sténopé, de la distance focale ainsi que de la distance séparant les lentilles, tel que décrit dans le modèle de calcul. Dans cette fonction, sympy est utilisé pour effectuer la résolution symbolique de la multiplication matricielle et des équations autant algébriques que matricielle. À la fin de cette fonction, les valeurs numériques sont substituées pour avoir une valeur numérique de résolution. Le programme utilise enfin matplotlib afin de visualiser le résultat de cette fonction pour des plages de paramètres.

Le graphique présenté en figure 1 démontre la variation de la résolution axiale pour une plage de diamètre de sténopé et ce, pour cinq valeurs différentes de longueur focale.

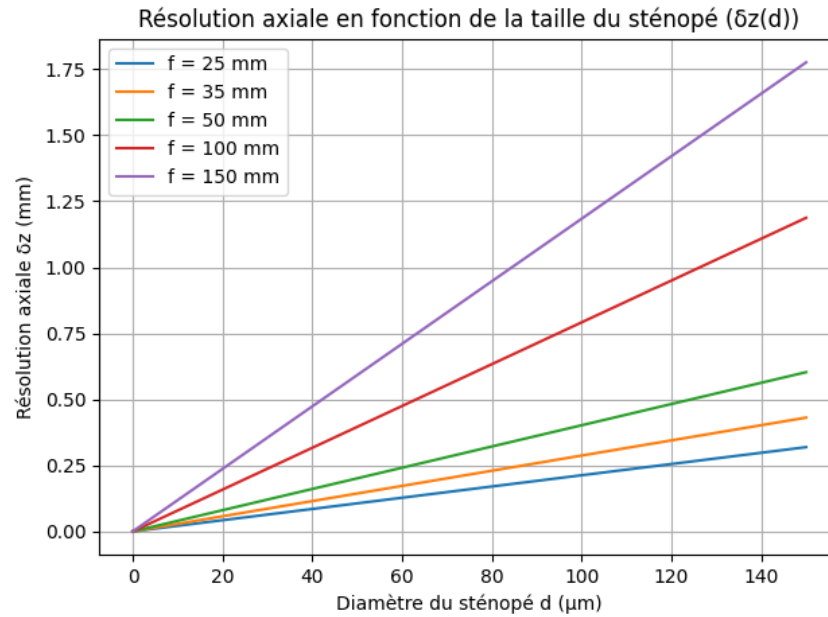


Figure 1 : Graphique de δ_z en fonction de d pour différentes valeurs de f .

” L’hypothèse présentée par le modèle de calcul est que la variation de d fait augmenter linéairement la résolution axiale. Ensuite, la figure 2 montre la variation de la résolution axiale pour une plage de longueur focale des deux lentilles, pour quatres valeurs différentes de diamètre de sténopé.

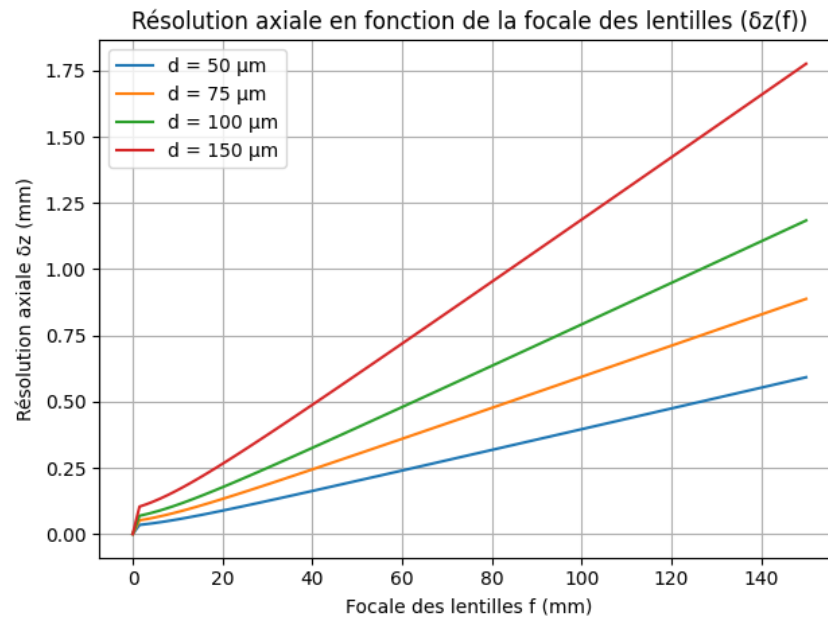


Figure 2 : Graphique de δ_z en fonction de f pour différentes valeurs de d .

Il est donc possible de prédire un comportement quasi-linéaire pour la résolution axiale en fonction de la focale. Ce comportement ne semble toutefois pas valide pour des valeurs de f très petites. Enfin, la figure

3 présente la résolution axiale en fonction de la distance entre les lentilles pour cinq valeurs différents de focales.

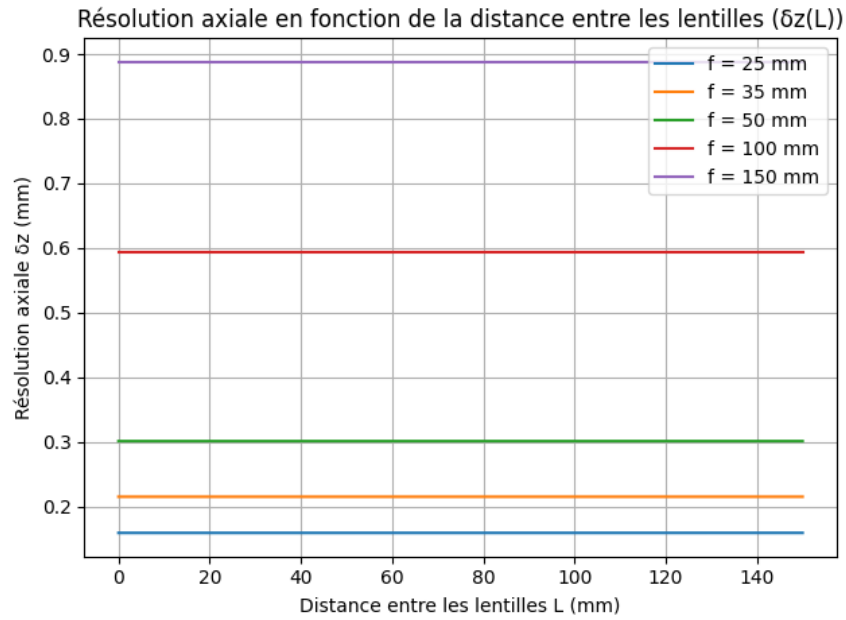


Figure 3 : Graphique de δ_z en fonction de L pour différentes valeurs de f .

L'hypothèse finale tirée du modèle est donc que la variation de L n'a pas d'impact sur la résolution.

5 Annexes

5.1 Modèle mathématique pour le calcul de résolution axiale

Soit le schéma de rayons du système optique présenté à la figure 4, qui provient de l'énoncé du laboratoire **(CITE!!!)** :

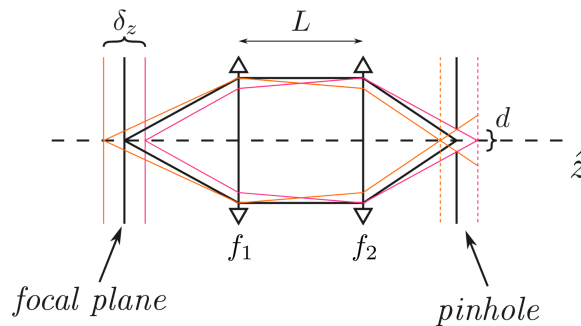


Figure 4 : Schéma illustrant les rayons passant aux extrémités du sténopé

Il est possible de mettre le *input plane* (IP) au plan focal, et le *output plane* (OP) au sténopé. Ainsi, il est possible de poser une matrice de transfert M au système de la façon suivante :

$$M = M_{t3}M_{f2}M_{t2}M_{f1}M_{t1} \quad (1)$$

où les matrices sont les suivantes :

- M_{t1} est la matrice de translation entre le plan focal et la lentille 1
- M_{f1} est la matrice de lentille mince de la lentille 2
- M_{t2} est la matrice de translation entre les deux lentilles
- M_{f2} est la matrice de lentille mince de la lentille 2
- M_{t3} est la matrice de translation entre le plan focal et la lentille 1

Une matrice de translation se décrit par :

$$M_t = \begin{bmatrix} 1 & x \\ 0 & 1 \end{bmatrix} \quad (2)$$

où x est la longueur de la translation selon l'axe optique. Une matrice de lentille mince se décrit par :

$$M_f = \begin{bmatrix} 1 & 0 \\ -\frac{1}{f} & 1 \end{bmatrix} \quad (3)$$

où f est la distance focale de la lentille. Soit r_i , un rayon de lumière initial au plan focal. Ce dernier se décrit par sa hauteur et par son angle par rapport à l'axe optique \hat{z} :

$$r_i = \begin{bmatrix} y_i \\ \alpha_i \end{bmatrix} = \begin{bmatrix} 0 \\ \alpha_i \end{bmatrix} \quad (4)$$

La hauteur y_i est nulle car le rayon de lumière commence sur l'axe optique. Soit aussi r_f , le rayon final qui arrive au sténopé :

$$r_f = \begin{bmatrix} y_f \\ \alpha_f \end{bmatrix} \quad (5)$$

C'est avec la définition de ce rayon final qu'il est possible de développer le modèle pour la résolution axiale δ_z . En effet, tel que visible à la figure 4, deux rayons finaux peuvent être dessinés à partir des deux extrémités du sténopé. Ces deux derniers convergent à des plans focaux qui ont une distance différente avec la première lentille. La différence entre ces distances est la résolution axiale recherchée.

Soit le cas 1, où l'on dénote la hauteur du rayon final au sténopé $y_{f1} = d/2$. Ce rayon est illustré en rouge. Il est possible de développer l'équation matricielle suivante :

$$r_{f1} = Mr_{i1} \quad (6)$$

$$\Rightarrow \begin{bmatrix} \frac{d}{2} \\ \alpha_{f1} \end{bmatrix} = M \begin{bmatrix} y_{i1} \\ \alpha_{i1} \end{bmatrix} \quad (7)$$

L'angle α_{f1} se trouve par trigonométrie :

$$\alpha_{f1} = \arctan\left(\frac{\phi_{f2} - d}{2f_2}\right) \quad (8)$$

où ϕ dénote le diamètre d'une lentille. Il est possible de dénoter la distance inconnue entre le plan focal du cas 1 et la lentille 1 par z_1 . Cette dernière se trouve dans M_{t1} , et donc dans M . Du coup, à l'aide de la librairie de calcul symbolique *sympy*, il est possible de calculer aisément la matrice de transfert et de résoudre l'équation (6) avec la méthode **LUSolve**. Le résultat a donc la forme suivante :

$$r_{i1} = \begin{bmatrix} y_{i1}(z_1) \\ \alpha_{i1}(z_1) \end{bmatrix} \quad (9)$$

Or, on sait que :

$$y_{i1}(z_1) = 0 \quad (10)$$

La fonction *sympy solve* permet de résoudre cette équation symboliquement pour z_1 , et après substitution, sa valeur numérique peut être connue.

Ces étapes de résolution se répètent pour le cas 2, où l'on dénote la hauteur du rayon final au sténopé $y_{f2} = -d/2$. Ce rayon est illustré en orange. Son angle se trouve également par trigonométrie :

$$\alpha_{f1} = \arctan\left(\frac{\phi_{f2} + d}{2f_2}\right) \quad (11)$$

Il est donc possible en résolvant les mêmes équations mais pour le cas 2 d'obtenir z_2 . Enfin, la résolution axiale se trouve ainsi :

$$\delta_z = |z_2 - z_1| \quad (12)$$

Cette valeur dépend donc de z , qui à son tour dépend des paramètres du montage physique.

5.2 Code source

```

1 import numpy as np
2 import matplotlib.pyplot as plt
3 import sympy as sp
4
5 def Mt(d):
6     '''Translation'''
7     M = np.array([[1, d],
8                   [0, 1]])
9     return M
10
11 def Ml(f):
12     '''Thin Lens'''
13     M = np.array([[1, 0],
14                   [-1/f, 1]])
15     return M

```

```

16
17 def res_axial(pinhole, focal, lens_distance):
18     """
19     Calcule la resolution axiale en fonction du diametre du trou, de la focale, et de la
20     distance entre les lentilles.
21
22     Parametres :
23     -----
24     pinhole : float
25         Diametre du trou en metres.
26     focal : float
27         Longueur focale de la lentille en metres.
28     lens_distance : float
29         Distance entre les lentilles en metres.
30
31     Retour :
32     -----
33     res : expression symbolique
34         La resolution axiale en metres.
35     """
36     f = sp.symbols('f')
37     phi = sp.symbols('phi')
38     d = sp.symbols('d')
39     L = sp.symbols('L')
40     z = sp.symbols('z')
41
42     # formation de la matrice de transfert
43     M = Mt(f)@Ml(f)@Mt(L)@Ml(f)@Mt(z)
44     M_sp = sp.Matrix(M)
45
46     # definition des hauteurs et angles de chaque cote du pinhole
47     yf1 = d/2
48     alphaf1 = -sp.atan((phi-d)/(2*f))
49
50     yf2 = -d/2
51     alphaf2 = -sp.atan((phi+d)/(2*f))
52
53     # vecteur decrivant le rayon final au pinhole
54     vf1 = sp.Matrix([yf1, alphaf1])
55     vf2 = sp.Matrix([yf2, alphaf2])
56
57     # resolution des systemes matriciels puis de leur 1re
58     # equation = 0 pour isoler z
59     solve1 = M_sp.LUsolve(vf1)
60     z1 = sp.solve(solve1[0], z)
61
62     solve2 = M_sp.LUsolve(vf2)
63     z2 = sp.solve(solve2[0], z)
64
65     # substitution des valeurs numeriques
66     values = {f:focal, phi:25.4e-3, d:pinhole, L:lens_distance}
67     res = z2[0].subs(values) - z1[0].subs(values)
68
69     return res
70
71 # set de courbes 1
72 focals = [25e-3, 35e-3, 50e-3, 100e-3, 150e-3]
73 pinhole_sizes = np.linspace(0, 150e-6, 100)
74 L = lambda f: f

```



```

74
75 plt.figure()
76
77 # Tracer une courbe pour chaque focale
78 for f in focals:
79     res_values = [res_axial(d, f, L(f)) * 1e3 for d in pinhole_sizes]
80     plt.plot(pinhole_sizes * 1e6, res_values, label=f'f = {f*1e3:.0f} mm')
81
82 # Parametres d'affichage
83 plt.xlabel("Diametre du stenope d (micro m)")
84 plt.ylabel("Resolution axiale delta z (mm)")
85 plt.legend()
86 plt.grid(True)
87 plt.title("Resolution axiale en fonction de la taille du stenope (dz(d))")
88 plt.tight_layout()
89 plt.savefig("res_vs_pinhole.png")
90 plt.show()
91 plt.clf()
92
93 # set de courbes 2
94 pinhole_sizes = [50e-6, 75e-6, 100e-6, 150e-6]
95 focals = np.linspace(1e-6, 150e-3, 100)
96 L = lambda f: f # si L(f) appelle, L = f
97
98 plt.figure()
99
100 # Tracer une courbe pour chaque taille de stenope
101 for pinhole in pinhole_sizes:
102     res_values = [res_axial(pinhole, f, L(f)) * 1e3 for f in focals]
103     plt.plot(focals * 1e3, res_values, label=f'd = {pinhole*1e6:.0f} micro m')
104
105 # Parametres d'affichage
106 plt.xlabel("Focale des lentilles f (mm)")
107 plt.ylabel("Resolution axiale dz (mm)")
108 plt.legend()
109 plt.grid(True)
110 plt.title("Resolution axiale en fonction de la focale des lentilles (dz(f))")
111 plt.tight_layout()
112 plt.savefig("res_vs_focal.png")
113 plt.show()
114 plt.clf()
115
116 # set de courbes 3
117 pinhole = 75e-6 # 75 micro m
118 focals = [25e-3, 35e-3, 50e-3, 100e-3, 150e-3] # en metres
119 lens_distances = np.linspace(0, 150e-3, 100) # en metres
120
121 plt.figure()
122
123 # Tracer une courbe pour chaque taille de focale
124 for f in focals:
125     res_values = [res_axial(pinhole, f, L) * 1e3 for L in lens_distances]
126     plt.plot(lens_distances * 1e3, res_values, label=f'f = {f*1e3:.0f} mm')
127
128 # Parametres d'affichage
129 plt.xlabel("Distance entre les lentilles L (mm)")
130 plt.ylabel("Resolution axiale dz (mm)")
131 plt.legend()
132 plt.grid(True)

```

```
133 plt.title("Resolution axiale en fonction de la distance entre les lentilles (dz(L))")
134 plt.tight_layout()
135 plt.savefig("res_vs_L.png")
136 plt.show()
137 plt.clf()
```