DM1 – Stigmatisme approché d'une lentille demi-boule

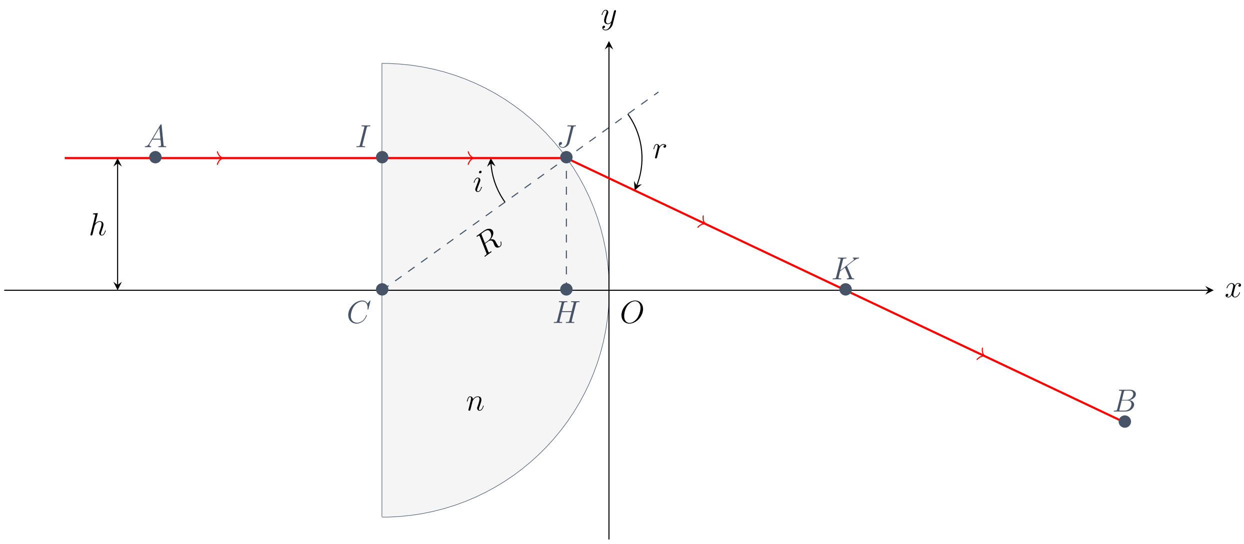
Objectif

→ Tester, à l'aide d'un langage de programmation, le stigmatisme approché d'une lentille demi-boule pour les rayons proches de l'axe optique.

Exercice 1 – Lentille demi-boule (26 points)

Le modèle des lentilles minces est une approximation des résultats de l'optique géométrique, qui reste valable tant que certaines conditions sont vérifiées. On souhaite explorer les limites de ce modèle à l'aide de Python.

On étudie pour cela la lentille demi-boule représentée ci-dessous. Elle est formée d'un milieu en verre d'indice n, délimité par deux dioptres : l'un plan et l'autre sphérique de rayon R et de centre C. La lentille est plongée dans l'air.



Étude théorique

On s'intéresse tout d'abord à un unique rayon incident représenté en rouge ci-dessus, parallèle à l'axe optique avant la lentille et repéré par sa distance à l'axe optique h.

1. Justifier que le rayon incident n'est pas dévié au niveau du point I.
2. Exprimer la relation entre les angles i et r à l'aide de la loi de Snell-Descartes.
3. Exprimer les distances algébriques $\overline{CH}$ et $\overline{HK}$ en fonction de i, r et R.
4. Montrer que $\overline{CK}$ s'exprime en fonction de i, R et n par la relation :

$$\overline{CK} = \frac{nR}{n\cos i - \sqrt{1 - n^2 \sin^2 i}}.$$

Prise en main du programme

Le programme `dioptre_spherique.py` permet de représenter la marche de quelques rayons arrivant sur le dioptre à différentes hauteurs h, issus d'une même source ponctuelle située à l'infini.

5. Donner l'indice du milieu choisi dans le programme.

6. Exécuter le programme `dioptre_spherique.py` (Ann. 1). Pourquoi les rayons les plus éloignés de l'axe optique n'émergent-ils pas de la lentille ?

Par la suite, on se restreindra aux cas où tous les rayons incidents émergent de la lentille.

7. Modifier et indiquer la plus grande valeur de `hmax` (ligne 9) qui permet de vérifier cette condition. On donnera la valeur avec deux chiffres significatifs.

8. Proposer un protocole permettant de déterminer le rôle de la variable `N` (ligne 10). Le mettre en œuvre en indiquant vos observations et donner le rôle de cette variable.

Stigmatisme approché

9. Justifier que la lentille demi-boule n'est pas un système optique stigmatique dans la situation considérée.

Il est possible d'obtenir un stigmatisme approché en limitant la partie utile de la lentille à un disque de rayon R_{D} à l'aide d'un diaphragme placé avant la lentille. Dans ce cas, l'image d'un objet ponctuel sera une tache de très faible taille et il sera possible d'obtenir des images nettes.

10. Indiquer le nom de la variable qui permet de reproduire l'effet d'un tel diaphragme.

11. Donner la valeur η_{max} du rapport $\eta = \frac{R_{\mathrm{D}}}{R}$ pour lequel le système vous parait stigmatique.

12. Proposer un critère (par exemple sous la forme d'une inégalité) pour rendre cette valeur moins arbitraire.

13. Donner un ou plusieurs inconvénients à l'utilisation d'un diaphragme très peu ouvert, c'est-à-dire pour lequel $\eta \ll 1$.

Conditions de Gauss

Les conditions de Gauss sont respectées quand les rayons lumineux sont :

- proches de l'axe optique ;
- peu inclinés par rapport à l'axe optique.

La plupart des systèmes optiques centrés sont approximativement stigmatiques dans ces conditions. Les rayons qui respectent les conditions de Gauss sont qualifiés de paraxiaux.

14. Préciser la première condition à l'aide d'une inégalité entre deux distances caractéristiques de la lentille.

Distance focale

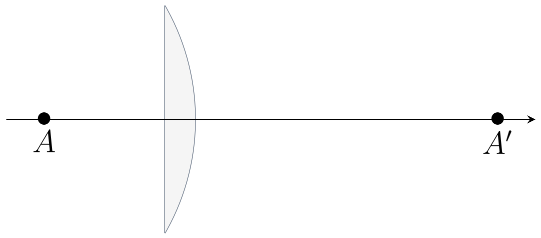
15. À l'aide du programme, indiquer la valeur de la distance focale f' de la lentille demi-boule, c'est-à-dire la distance $\overline{OK}$.

16. Commenter cette valeur par rapport au résultat de la question 4.

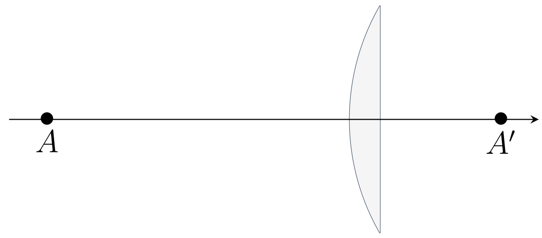
Règle des 4P

Pour obtenir une image la plus nette possible, il est recommandé d'appliquer la règle du « plus plat plus près », ou règle des 4P. Elle consiste à orienter la face la plus plate d'une lentille :

- du côté de l'objet si celui-ci est plus proche de la lentille que son image :



- du côté de l'image si celle-ci est plus proche de la lentille que l'objet :



Le programme `regle_4p.py` (Ann. 2) permet d'étudier simultanément les deux configurations. Les noms de variable sont les mêmes que dans le programme précédent.

17. Reprendre la question 11 dans le cas de la lentille inversée.

18. Comparer les valeurs obtenues dans les deux cas et commenter la règle des 4P.

Annexes

Annexe 1 – dioptre_spherique.py

```python
import numpy as np
import matplotlib.pyplot as plt

############
# PARAMÈTRES
############
R = 1          # rayon du dioptre sphérique (en mètres)
n = 1.5        # indice de la lentille
hmax = 0.7     # distance maximale du rayon incident (en mètres)
N = 6

##########################
# PRÉPARATION DE LA FIGURE
##########################
fig = plt.figure(figsize=(12,8))
plt.plot([-2*R, 5*R], np.zeros(2), "--k") # axe optique
fig.axes[0].set_aspect("equal") # même échelle sur les deux axes
plt.xlim(-2*R, 5*R)
plt.xlabel("x (m)")
plt.ylabel("y (m)")
# Dioptre sphérique
theta = np.linspace(-np.pi/2, np.pi/2, 101)
xd = R * np.cos(theta) - R
yd = R * np.sin(theta)
plt.plot(xd, yd, "k")
plt.plot([xd[0], xd[-1]], [yd[0], yd[-1]], "k")

#################
# TRACÉ DE RAYONS
#################
for h in np.linspace(-hmax, hmax, N):
    # Coordonnées des différents points
    xA, yA = (-2 * R, h)                          # coordonnées du point A
    xI, yI = (-R, h)                              # coordonnées du point I
    xJ, yJ = (np.sqrt(R**2 - h**2)-R, h) # coordonnées du point J
    # Angles au niveau du dioptre sphérique
    i = np.arcsin(h / R)       # angle d'incidence
    r = np.arcsin(n * h / R) # angle de réfraction
    xK, yK = (xJ + h / np.tan(r-i), 0)    # coordonnées du point K
    xB, yB = (2*xK-xJ, -h)                        # coordonnées du point B
    # Tracé de la marche des rayons
    x = [xA, xI, xJ, xK, xB]
    y = [yA, yI, yJ, yK, yB]
    plt.plot(x, y, "red")
plt.show()
```

Annexe 2 – `regle_4p.py`

```
import numpy as np
import matplotlib.pyplot as plt

############
# PARAMÈTRES
############
R = 1           # rayon du dioptre sphérique (en mètres)
n = 1.5         # indice de la lentille
hmax = 0.7      # distance maximale du rayon incident (en mètres)
N = 6

##########################
# PRÉPARATION DE LA FIGURE
##########################
plt.figure(figsize=(16,12))
ax1 = plt.subplot2grid((2,1), (0,0))
ax2 = plt.subplot2grid((2,1), (1,0))
ax1.set_title("Configuration initiale")
ax2.set_title("Configuration inversée")
for ax in [ax1, ax2]:
    ax.set_aspect("equal")
    ax.set_xlim(-2*R, 3*R)
    ax.plot([-2*R, 5*R], np.zeros(2), "--k") # axe optique
    ax.set_xlabel("x (m)")
    ax.set_ylabel("y (m)")
# Dioptre sphérique (sens ordinaire)
theta = np.linspace(-np.pi/2, np.pi/2, 101)
xd = R * np.cos(theta) - R
yd = R * np.sin(theta)
ax1.plot(xd, yd, "k")
ax1.plot([xd[0], xd[-1]], [yd[0], yd[-1]], "k")
# Dioptre sphérique (sens inversé)
theta = np.linspace(-np.pi/2, np.pi/2, 101)
xd = -R * np.cos(theta)
yd = R * np.sin(theta)
ax2.plot(xd, yd, "k")
ax2.plot([xd[0], xd[-1]], [yd[0], yd[-1]], "k")

#################
# TRACÉ DE RAYONS
#################
# Configuratiuon initiale
for h in np.linspace(-hmax, hmax, N):
    # Coordonnées des différents points
    xA, yA = (-2 * R, h)                  # coordonnées du point A
    xI, yI = (-R, h)                      # coordonnées du point I
    xJ, yJ = (np.sqrt(R**2 - h**2)-R, h) # coordonnées du point J
    # Angles au niveau du dioptre sphérique
```

```
    i = np.arcsin(h / R)      # angle d'incidence
    r = np.arcsin(n * h / R) # angle de réfraction
    xK, yK = (xJ + h / np.tan(r-i), 0)    # coordonnées du point K
    xB, yB = (2*xK-xJ, -h)                # coordonnées du point B
    # Tracé de la marche des rayons
    x = [xA, xI, xJ, xK, xB]
    y = [yA, yI, yJ, yK, yB]
    ax1.plot(x, y, "red")
# Configuration inversée
for h in np.linspace(-hmax, hmax, N):
    # Coordonnées des différents points
    xA, yA = (-2 * R, h)                  # coordonnées du point A
    xI, yI = (-np.sqrt(R**2 - h**2), h)   # coordonnées du point I
    # Angles au niveau des dioptres
    alpha = np.arctan(h / R)
    i1 = np.arcsin(np.sin(alpha)/n)
    i2 = alpha - i1
    i3 = np.arcsin(n*np.sin(i2))
    x0 = - np.sqrt(R**2 - h**2)
    y0 = h - (R - x0) * np.tan(i2)
    L = y0 / np.tan(i3)
    xJ, yJ = (0, y0)                      # coordonnées du point J
    xK, yK = (L, 0)                       # coordonnées du point K
    xB, yB = (2*L, -y0)                   # coordonnées du point B
    # Tracé de la marche des rayons
    x = [xA, xI, xJ, xK, xB]
    y = [yA, yI, yJ, yK, yB]
    ax2.plot(x, y, "red")
plt.show()
```