

 Unité de formation des Sciences de l'Ingénieur	ANNEE UNIVERSITAIRE 2016-2017 Mention : Sciences pour l'ingénieur UE : 4TSI402U Epreuve : Examen d'informatique modulaire Date : Mercredi 3 mai 2017 Heure : 14h30 Durée : 1h30 Tout document papier autorisé
--	---

1 Lentille simple

On se propose de résoudre numériquement un problème d'optique simple : le calcul de la position et de la taille de l'image d'un objet par une lentille mince convergente par une approche géométrique.

Données du problème : On considère une lentille mince convergente de distance focale f centrée perpendiculairement à l'axe optique au point O . Le problème est de calculer la position et la taille de l'image qu'elle donne d'un objet AB , placé lui aussi perpendiculairement à l'axe optique, le point A étant sur l'axe optique.

Résolution par construction des rayons : Principe de résolution

La résolution du problème peut se faire en traçant quelques rayons caractéristiques issus de B , puisque tout rayon issu de B converge en B' point conjugué de B par la lentille. L'image A' de A étant bien évidemment sur l'axe, $A'B'$ restant perpendiculaire à celui-ci.

Règles de construction

1. tout rayon parallèle à l'axe optique émerge de la lentille en passant par le foyer image de celle-ci.
2. tout rayon passant par le centre de la lentille émerge de celle-ci sans être dévié.
3. tout rayon passant par le foyer objet de la lentille émerge parallèlement à l'axe optique.

On trace deux rayons caractéristiques issus de B , les rayons émergeant de la lentille se coupent en B' . La figure 1 présente cette construction pour trois rayons caractéristiques. L'algorithme utilisé dans le programme ne tient compte que des rayons (1) et (2).

Solution du problème Afin d'obtenir la position de l'objet et sa taille, il suffit de calculer les coordonnées du point B' .

L'objectif de ce problème est d'écrire un programme permettant de calculer les caractéristiques de l'image (position et taille) en fonction des caractéristiques de l'objet (position et taille) et des caractéristiques de la lentille (position et longueur focale), par application des règles ci-dessus¹.

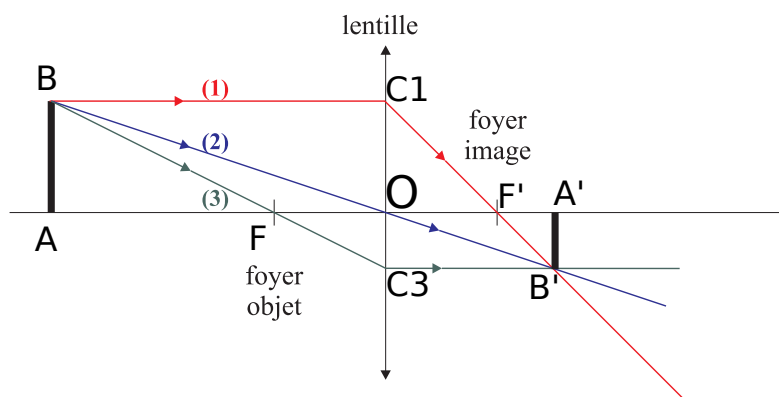


FIGURE 1 – Principe de calcul de l'image générée par une lentille

1. Il existe des formules beaucoup plus simples qui découlent de ce raisonnement. Nous ne les utiliserons pas.

Le code est constitué de 5 fichiers : `geometrie.c`, `geometrie.h`, `optique.c`, `optique.h`, `test_optique.c`. Une partie du code du programme est donnée en annexe. Certaines fonctions sont incomplètes. Les compléter fera l'objet du problème.

Un exemple d'exécution du programme (complété) donne (les réponses de l'utilisateur sont en **gras**) :

Quelle sont la position (x) et la longueur focale de la lentille? **10 30**

Quelle sont la position (x) et la taille (y) de l'objet? **-80 20**

Position de l'image : 55.000000 Taille de l'image : -10.000000

- 1) Quelle(s) ligne(s) de commande faut-il taper dans un terminal pour compiler le programme (en admettant qu'il a été convenablement complété). On pourra utiliser une seule ou plusieurs lignes de commande pour la compilation.

Module geometrie

- 2) Que signifient les lignes 5 à 8 de `geometrie.h` ?
- 3) Que signifie la ligne 9 de `test_optique.c` ?
- 4) Cinq fonctions sont déclarées dans `geometrie.h`. Dans quel fichier doit-on écrire les définitions correspondantes ?
- 5) Écrivez le code (définition) de la fonction `point_cree` qui crée un point à partir de ses coordonnées.
- 6) Écrivez le code de la fonction `droite_calculer_intersection` qui calcule l'intersection de deux droites.
- 7) Écrivez le code de la fonction `droite_cree_par2points` qui calcule la droite passant par deux points.

Remarque : le choix de la représentation d'une droite par l'équation à trois paramètres a été fait pour autoriser les droites verticales et horizontales. Par conséquent, une même droite peut avoir plusieurs équations différentes, dont les coefficients sont proportionnels. Le degré de liberté restant est laissé au choix du programmeur du module. La simplicité est recommandée.

- 8) Écrivez le code de la fonction `droite_cree_parallele` qui calcule la droite parallèle à une droite donnée, passant par un point donné.

Module optique

Le type `Lentille` est déclaré dans `optique.h` ainsi que les fonctions associées.

- 9) Dans le corps de la fonction `lentille_calculer_image` (donné à la fin de l'annexe), que signifie le symbole : `->` dans `lentille->centre`. Par quoi peut-on le remplacer ?
- 10) Écrivez le code complet de `optique.h`.

Pour vous guider :

- A partir du code de la fonction `lentille_calculer_image`, identifiez les champs du type `Lentille`.
- A partir du code de la fonction `main`, identifiez les déclarations des fonctions du module `optique`.

2 Système optique

Le programme précédent marchant parfaitement, on décide d'étendre ses possibilités pour pouvoir calculer les caractéristiques d'un système optique (ce que l'on appelle plus brièvement une optique) constitué de plusieurs lentilles alignées (par exemple un microscope ou une lunette).

L'optique est représentée par une liste correctement ordonnée de lentilles.

Pour représenter une optique, on déclare :

```
define NB_LENTILLES_MAX 10
```

```
typedef struct Optique {
    int nb_lentilles;
    Lentille lentilles[NB_LENTILLES_MAX];
} Optique;
```

Pour calculer l'image d'un objet par une optique, on calcule l'image de l'objet par la première lentille, puis l'image de la première image par la deuxième lentille etc. . . L'image par l'optique est l'image obtenue par la dernière lentille (voir figure 2)

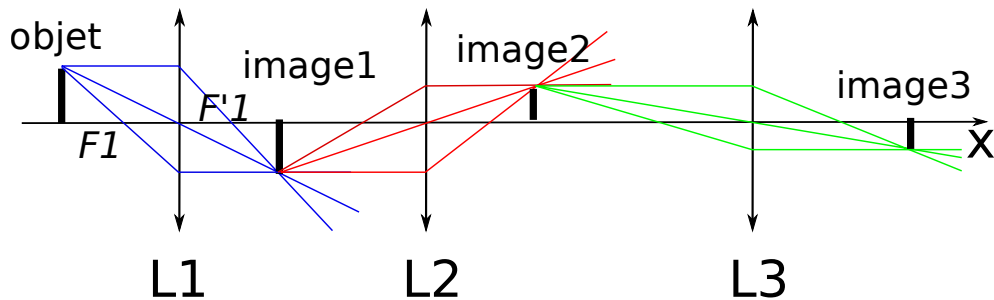


FIGURE 2 – Principe de calcul de l'image générée par une optique constituée de 3 lentilles

- 11) Écrire la fonction `Point optique_calculer_image(Optique *optique, Point objet)` ; qui calcule l'image de l'objet par l'optique. On supposera que les lentilles sont rangées dans le bon ordre au sein de la variable `optique`.

3 ANNEXES : codes

```
1 /* geometrie.h */
2 #ifndef GEOMETRIE_H
3 #define GEOMETRIE_H
4
5 typedef struct Point {
6     double x;
7     double y;
8 } Point;
9
10 typedef struct Droite {
11     double a;          /*eq : a * x + b * y + c = 0 */
12     double b;
13     double c;
14 } Droite;
15
16 Point point_creer(double x, double y);
17
18 Droite droite_creer_par2points(Point M, Point N);
19 Droite droite_creer_parallele(Droite D, Point P);
20 Droite droite_creer_perpendiculaire(Droite D, Point P);
21
22 Point droite_calculer_intersection(Droite D1, Droite D2);
23
24 #endif
```

```

1 /* test_optique.c */
2 #include <stdio.h>
3 #include "optique.h"
4 #include "geometrie.h"
5
6 int main(void)
7 {
8     double position, longueur_focale;
9     Point objet, image;
10    Lentille lentille;
11
12    printf("Quelle sont la position (x) et la longueur focale de la lentille ?");
13    scanf("%lf%lf",&position, &longueur_focale);
14    lentille=lentille_creer(position,longueur_focale);
15
16    printf("Quelle sont la position (x) et la taille (y) de l'objet ?");
17    scanf("%lf%lf", &objet.x,&objet.y);
18
19    image=lentille_calculer_image(&lentille,objet);
20
21    printf("Position de l'image : %lf    Taille de l'image : %lf\n",
22          image.x,image.y);
23
24    return 0;
25 }

```

Corps de la fonction lentille_calculer_image

```

1 {
2     Droite D1,D2,B_C1;
3     Point image, C1;
4
5     D2 = droite_creer_par2points(objet,lentille->centre);
6
7     B_C1 = droite_creer_parallele(lentille->axe, objet);
8     C1 = droite_calculer_intersection(lentille->plan_lentille, B_C1);
9     D1 = droite_creer_par2points(C1, lentille->foyer_image);
10
11    image = droite_calculer_intersection(D1,D2);
12
13    return image;
14 }

```

FIN