Une trajectoire elliptique

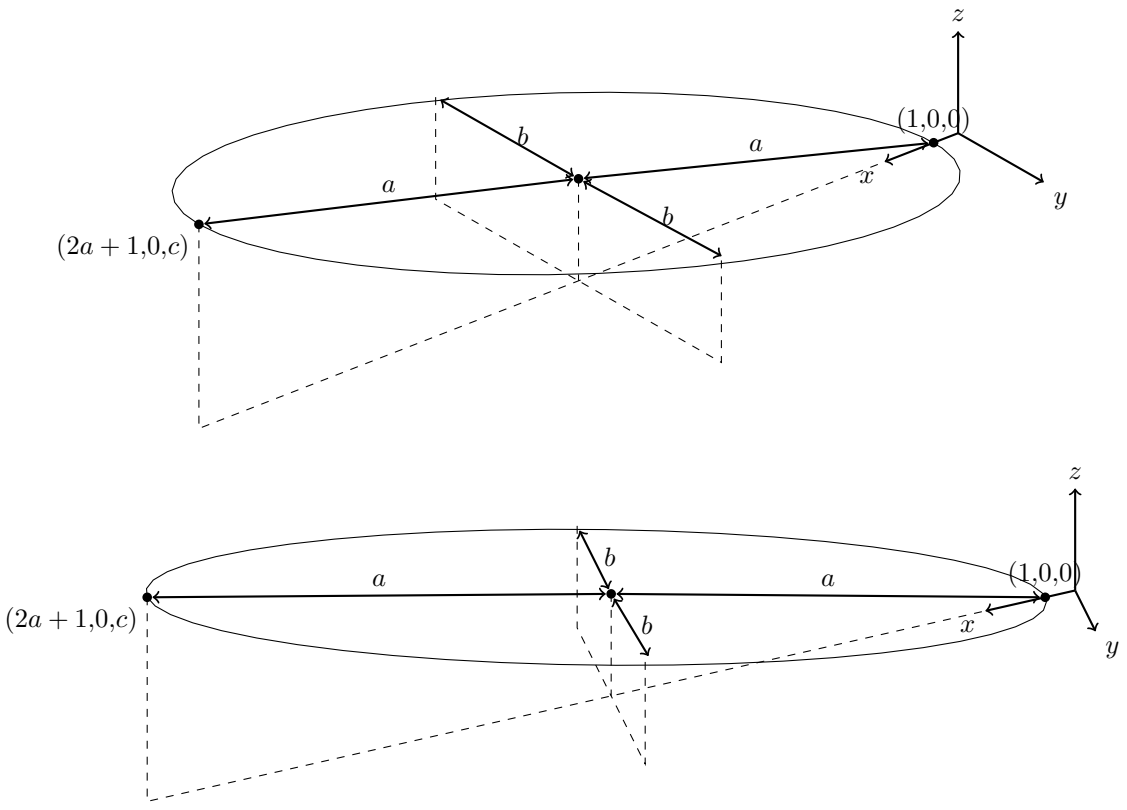
- Travail à effectuer obligatoirement en **équipe** de 2 ou 3 personnes.
- Pondération : jusqu'à 10 points ajoutés à la note du devoir.
- Les équipes ne sont pas forcément les mêmes que pour le devoir.
- Remise par courriel à l'adresse `xavier.provencal@etsmtl.ca` au plus tard **48 heures après la fin de l'atelier**.
- La note de zéro sera attribuée à tout travail remis en retard.
- Lorsque vous rendez votre travail, n'envoyez **que le fichier source utilisé**.

Objectif : Réaliser un programme qui génère la liste des prises de vues nécessaires pour réaliser un film où la caméra effectue une trajectoire elliptique décrite par les équations paramétriques :

$$\begin{aligned} x(t) &= a\,(1+\cos(t)) + 1 \\ y(t) &= b\,\sin(t) \qquad\qquad , \quad 0 \le t \le 2\pi \\ z(t) &= c\,(1+\cos(t)) \end{aligned}$$

Note :

- Les extrémités de l'ellipse sont (1,0,0) et $(2a+1,0,c)$.
- a est la longueur du demi-axe aligné avec l'axe des x.
- b est la longueur du demi-axe aligné avec l'axe des y.
- c est la hauteur (la valeur en z) du point le plus éloigné de l'origine.
- Consultez l'adresse `http://xprov.org/film_q4.mp4` pour visualiser le résultat attendu avec $a = 15$, $b = 5$ et $c = 3$. Un tel film devrait être obtenu à la question 4.



Consignes

1. Le programme ne doit **jamais** interagir avec l'utilisateur.
2. Retirez ou commentez tout affichage superflu.
3. Consultez la page Moodle du cours pour la mise en place de cet atelier.

Q0 Prise en main
Prenez connaissance du fichier source `trajectoire.c/cpp/java/py` (celui de votre choix). En particulier, lisez attentivement la fonction `question0`. Exécutez le programme en redirigeant la sortie standard dans le fichier `cameras.json`. Exécutez le programme **Visu3d** et vérifiez que le film produit correspond bien à la trajectoire décrite en commentaire de la fonction `question0`.

Q1 Réalisez la trajectoire **(2.5 points)**
Inspirez-vous du code de la fonction `question0` pour écrire la fonction `question1` telle que :

- La signature de la fonction est la même que pour `question0`,
- Le paramètre n est le nombre d'images, soit le nombre de fois que la `camera` est affichée.
- Les positions des caméras doivent décrire exactement un tour de la trajectoire $\vec{r}(t) = (x(t), y(t), z(t))$.
- Le vecteur `orientationRegard` est initialisé à $(-1,0,0)$ et n'est jamais modifié.
- Le vecteur `orientationVersLeHaut` est initialisé à $(0,0,1)$ et n'est jamais modifié.
- Pour obtenir une variable t qui varie de 0 à 2π en n pas égaux faites :

```
Pour i de 0 à n-1 faire
     t = (2 π i) / (n-1)
```

- Définissez trois constantes : $a = 15.0$, $b = 5.0$ et $c = 3.0$. Ces constantes sont les paramètres de l'ellipse tels qu' illustrés à la page précédente. Votre code doit fonctionner peu importe la valeur de ces constantes.
  Remarque : on les supposent strictement positives, ne faites pas de validation pour cela.

Q2 Regarder dans la direction du mouvement **(2.5 points)**
Faites un copier/coller de la fonction écrite à la question précédente et renommez-la `question2`. Modifiez le code de cette fonction de manière à ce que pour chaque position le vecteur `orientationRegard` pointe dans direction du mouvement.

- *Regarder dans la direction du mouvement* signifie que l'orientation de la caméra est alignée avec la tangente à la courbe dans le sens positif du mouvement.
- Si vous ne voyez pas de quoi il est question, relisez la section 10.2 du Stewart.
- Votre vecteur n'a pas besoin d'être unitaire.

Attention : lorsqu'on modifie le vecteur `orientationRegard` de la caméra, il faut également modifier le vecteur `orientationVersLeHaut` car les deux doivent toujours être perpendiculaires. Pour tout de suite, on se contente d'obtenir un vecteur perpendiculaire à $\vec{v} = (x,y,z)$ en faisant :

$$\vec{v} \wedge \vec{i} = (x,y,z) \wedge (1,0,0) = (0,z,-y).$$

Q3 Vecteur normal principal **(2.5 points)**
Faites un copier/coller de la fonction écrite à la question précédente et renommez-la `question3`. Modifiez le code de cette fonction de manière à ce que pour chaque position le vecteur `orientationVersLeHaut` pointe dans la direction du **vecteur normal principal**.

- Le vecteur normal principal est défini comme étant la dérivée du vecteur tangent unitaire.
- Si vous ne voyez pas de quoi il est question, référez-vous à la page 712 du Stewart.
- Votre vecteur n'a pas besoin d'être unitaire.

Q4 Regarder "Vers le haut" **(2.5 points)**
Faites un copier/coller de la fonction écrite à la question précédente et renommez-la `question4`. Modifiez le code de cette fonction de manière à ce que pour chaque position le vecteur `orientationVersLeHaut` pointe dans une direction qui est perpendiculaire vecteur normal principal.