



IUT - Département Informatique  
*LP Dawin Alt - WebGL - 2017-2018*  
**Semaine 5 - Devoir Machine**

# WebGL Devoir Machine

## Consignes

Durée de l'épreuve : 2h.  
Le barème est donné à titre indicatif. Le travail produit doit être remis sur Moodle, module WebGL, sous la forme d'une archive .zip ou .tar.gz.

Ce travail sera effectué à l'aide de la bibliothèque Three.js. On rappelle que la bibliothèque peut être récupérée à l'adresse <https://raw.githubusercontent.com/mrdoob/three.js/master/build/three.min.js>. La documentation est accessible à l'adresse : <https://threejs.org/docs/>.

Commencez par récupérer et lire le code contenu dans l'archive. Les parties à compléter se situent uniquement dans le fichier javascript et sont marquées d'un commentaire "TODO".

## Modélisation du rebond d'une balle

Le but de ce travail est de modéliser le rebond d'une balle élastique sur un sol. Le rebond se compose de deux phases :

1. **Phase de descente** : chute libre de la balle à partir d'une altitude donnée, jusqu'à ce que la balle atteigne le sol ;
2. **Phase d'ascension** : rebond de la balle jusqu'à une fraction de l'altitude initiale.

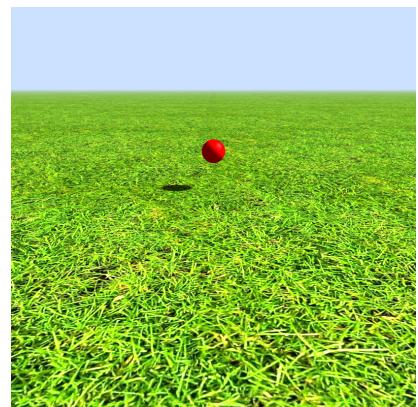
Ces différentes phases sont décrites par des équations de la physique du mouvement. L'objectif est ici de proposer un rendu réaliste et qui soit donc proche de la physique du rebond.



(a) Phase de descente



(b) Rebond sur le sol



(c) Phase d'ascension

## 1 Création d'une scène minimale

(/4 points)

Le but de cette partie est de créer les deux objets principaux de la scène : une balle, ainsi que le sol. Compléter la fonction `main()`.

## 1.1 Création de la balle

1. Créez une sphère avec pour valeur de rayon la variable globale `radius`.
2. Positionnez-la en  $y = altitudeMax$ .
3. Colorez-la en rouge.

## 1.2 Création du sol

1. Créez le sol : on considère ici qu'il est plan. Aidez-vous de la documentation afin de trouver la géométrie correspondante.
2. Positionnez-le à l'horizontale, en  $y = 0$ .
3. Colorez-le en vert.

# 2 Implémentation du rebond de la balle sur le sol

(/12 points)

Le but de cette partie est d'implémenter la chute libre de la balle, le rebond de celle-ci sur le sol et la phase d'ascension qui s'ensuit, ainsi que la déformation de la balle au moment du choc sur le sol (voir Fig.1b).

On cherche ici à faire une **animation**, dans le sens où il s'agit d'un processus qui fait varier la position de la balle au cours du temps. En WebGL, on effectue une animation par la méthode `requestAnimationFrame()` qui permet d'appeler une fonction de dessin en callback. La fonction `animationRebond()` correspond au code à produire pour modéliser le rebond. Le temps y est matérialisé par la variable `t` et incrémenté à chaque nouveau dessin.

## 2.1 Chute libre

**Préambule** La position d'un objet en chute libre à un instant  $t$  et à une altitude initiale  $y_0$  est modélisée par l'équation :

$$y(t) = y_0 - \frac{1}{2} * g * t^2$$

où  $g$  est une constante qui représente l'accélération de la pesanteur terrestre ( $= 9.81m/s^2$ )

### Questions

1. Compléter la fonction `chuteLibre()` de façon à ce qu'elle retourne la position en  $y$  de la balle en fonction du temps.
2. Dans la fonction `animationRebond()`, mettre à jour la position de la sphère avec la valeur renournée par la fonction `chuteLibre()`.

## 2.2 Phase d'ascension et rebond

**Préambule** A ce stade, seule la phase de descente de la balle est modélisée. L'objectif est ici de gérer le rebond de la balle sur le sol. La vitesse de la balle lancée après le  $n$ -ième rebond à partir d'une altitude initiale  $y_0$  est donnée par :

$$v = E^n * \sqrt{2 * g * y_0}$$

où  $E$  est le coefficient d'élasticité de la balle. Un coefficient  $E = 1$  signifie que la balle est très élastique et qu'elle ne dissipe pas son énergie. La balle va ainsi remonter à son altitude initiale  $y_0$ . On pourra choisir  $E = 0.9$  dans la suite du travail.

La position d'un objet à un instant  $t$  est donnée par :

$$y(t) = v * t - \frac{1}{2} * g * t^2$$

Cette équation décrit à la fois la phase d'ascension **et** la phase de descente de la balle.

**NB** Ces équations tiennent si la balle heurte le sol à  $t = 0$ .

### Questions

1. Compléter la fonction `rebondBalle()`, de façon à ce qu'elle retourne la position en  $y$  de la balle en fonction du temps, après un rebond.
2. Dans la fonction `animationRebond()`, à chaque fois que la balle heurte le sol : décomptez le nombre de rebonds ET réinitialisez le temps à 0.
3. Dans la fonction `animationRebond()`, mettre à jour la position de la sphère avec la valeur renournée par la fonction `rebondBalle()`. Faites en sorte que l'animation commence avec la balle à sa hauteur initiale ( $y = 100$ ).

## 2.3 Déformation de la balle

**Préambule** Le but de cette partie est de déformer la balle à chaque rebond (voir Fig. 1b), afin de réaliser un rendu "élastique" de la balle. La déformation correspond à un aplatissement de la balle dans la direction de la chute. Cet aplatissement peut être modélisé par l'attribut `scale` de `THREE.Mesh`, qui permet d'allonger en  $x$ ,  $y$ , et  $z$  les dimensions d'un objet.

**Question** Dans la fonction `deformation()`, proposez votre propre modèle de déformation au cours du temps en modifiant l'attribut `scale` de la sphère. Explicitez votre modèle en commentaire.

## 2.4 Trajectoire parabolique

**Préambule** On cherche désormais à modéliser la trajectoire d'un lancer de balle. Cela signifie qu'on souhaite modifier la position en  $x$  de la balle au cours du temps, sans modifier la position en  $y$ . Un modèle simple considère que le déplacement en  $x$  est constant.

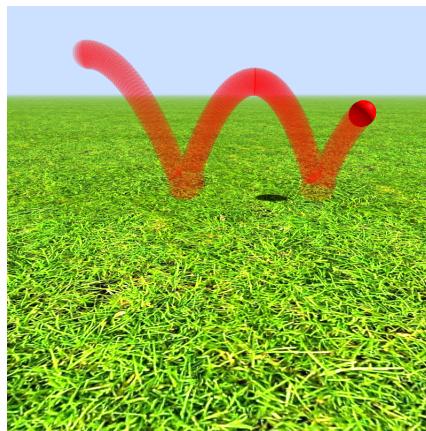


FIGURE 2 – Trajectoire parabolique de la balle.

### Questions

1. Modifier la fonction `animationRebond()` afin de modéliser une trajectoire parabolique de la balle, en fixant un déplacement en  $x$  constant au cours du temps. **NB** : La position de la sphère en  $y$  reste inchangée par rapport aux questions précédentes.
2. Quel est le problème d'une telle approche ? Proposer un modèle de déplacement qui y apporte une solution. Explicitez votre modèle en commentaire.

## 3 Vers un rendu plus réaliste de la scène

(/4 points)

**Questions** L'objectif de cette partie est de s'approcher d'un rendu réaliste, comme montré sur les Figures 1a à 1c.

1. Modéliser un sol herbeux à l'aide de l'image de texture contenue dans le répertoire `textures`.
2. Ajouter une source lumineuse ponctuelle à la scène. La balle doit avoir une ombre propre et une ombre portée sur le sol.