Groupe 13 PELUSO Nathan – DECENCIERE-FERRANDIERE Luis

Rapport de projet INF443

Kite-Surf

Dans ce projet d’informatique graphique, nous avons décidé de travailler sur une scène simple mais mettant en œuvre de nombreuses techniques et méthodes : l’animation d’un personnage, tiré par une voile, sur un océan à la surface variable au cours du temps.

Table des matières

[I) Le surfeur 2](#_Toc73439345)

[a. Hiérarchie 2](#_Toc73439346)

[b. Cinématique inverse 3](#_Toc73439347)

[II) Meshes 4](#_Toc73439348)

[a. Planche 4](#_Toc73439349)

[b. Voile 5](#_Toc73439350)

[III) Océan 5](#_Toc73439351)

[a. Forme, paramètres 5](#_Toc73439352)

[b. Ajouts 5](#_Toc73439353)

[IV) Simulation physique 6](#_Toc73439354)

[a. Voile 6](#_Toc73439355)

[b. Corde 6](#_Toc73439356)

## Le surfeur

### Hiérarchie

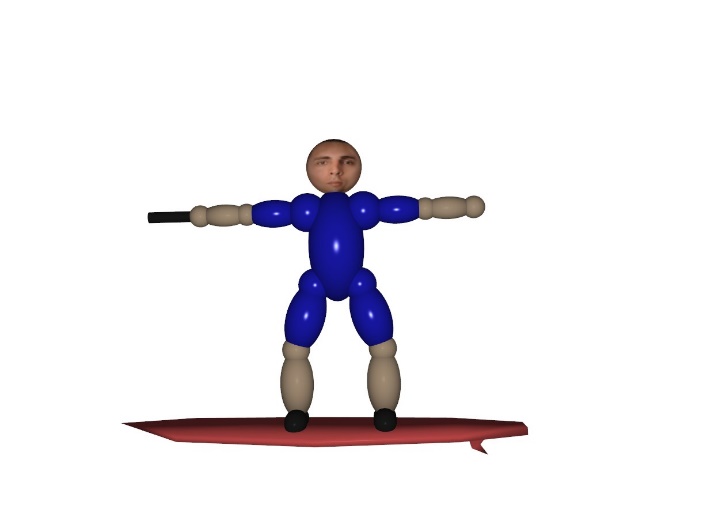
Ci-après est schématisée la hiérarchie utilisée pour définir et articuler notre personnage surfeur. En vert les objets sphériques, en jaune les ellipsoïdes, et en bleu les maillages créés manuellement.

La voile n’appartient pas à la hiérarchie, mais est représentée ici en tant que maillage créé manuellement.

Ces objets sont ensuite texturés (pour la tête) ou simplement colorés en utilisant les shaders par défaut, et en jouant sur les paramétrages de l’ombrage de Phong pour obtenir des effets de texture différents.



*Position du surfeur pendant la simulation, ici jambes tendues*



*Position neutre du surfeur*

### Cinématique inverse

Il s’agit à présent d’animer le surfeur, étant donné une fonction *altitude(x,y,t)* et une position *p={x,y,z}(t)*, pour qu’il soit dans une position réaliste. Pour ce faire, nous avons dû travailler en cinématique inverse :

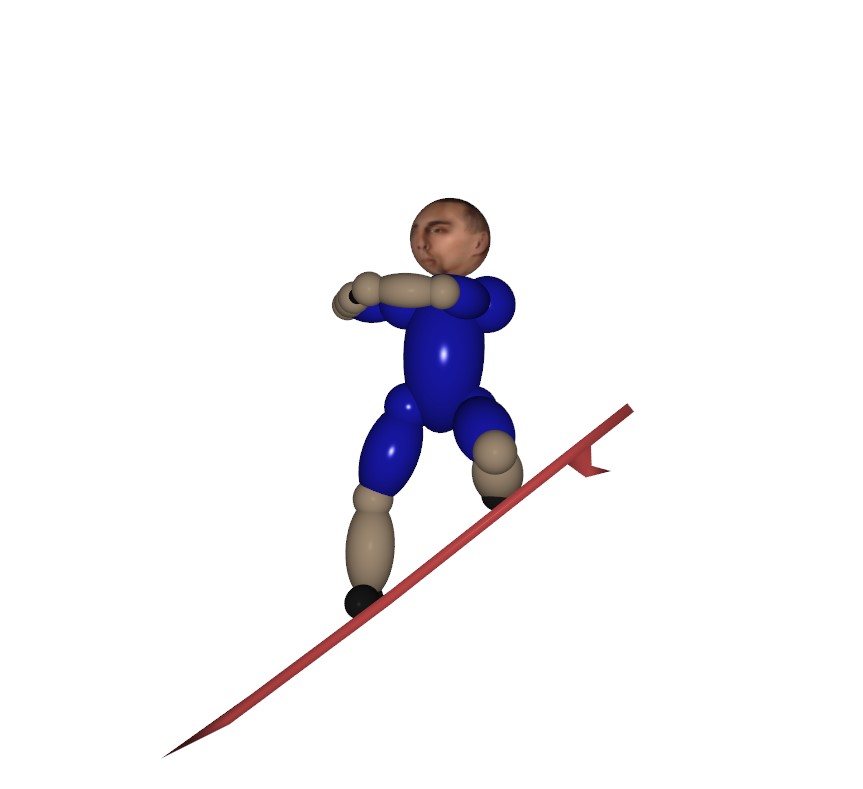
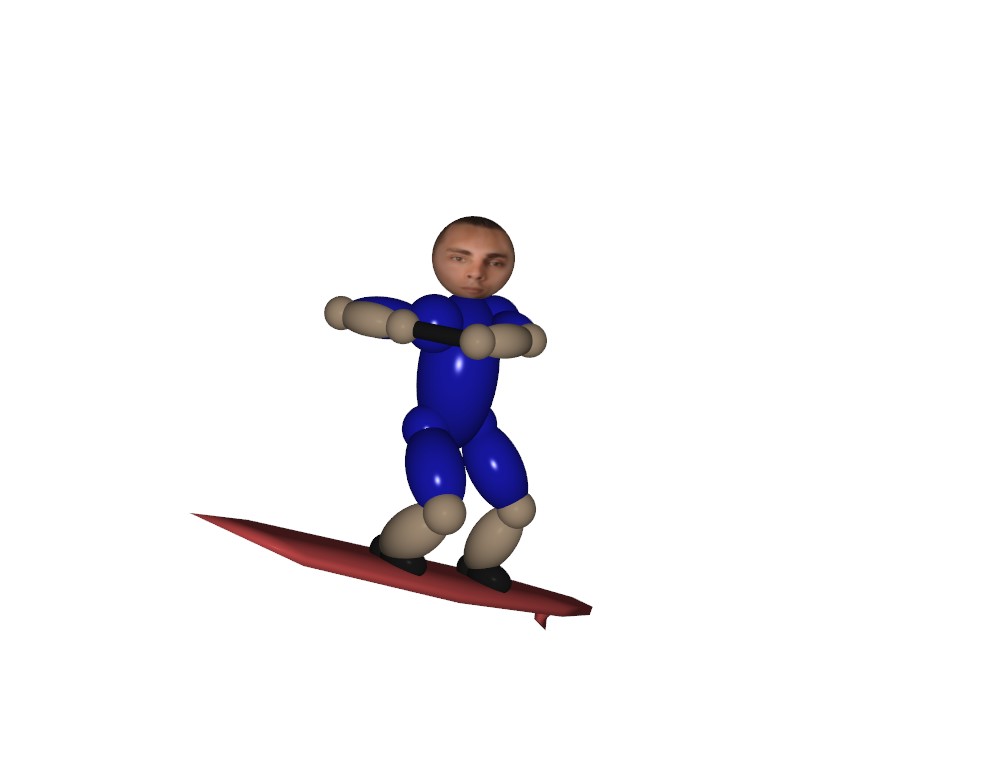
* Une image contenant texte

  Description générée automatiquementTout d’abord, on place le surfeur à la position déterminée par notre simulation physique, et l’on utilise son vecteur vitesse pour l’aligner (calcul d’un angle de lacet).
* Une image contenant texte

  Description générée automatiquementEnsuite il s’agit de plier les jambes indépendamment afin d’incliner la planche de surf pour qu’elle s’aligne avec l’eau, tout en s’assurant que les deux pieds restent en contact avec celle-ci. Pour cela, on calcule l’altitude de l’eau sous les pieds avant (droit) et arrière(gauche), et on calcule les différents angles autour de la position médiane « jambes semi-fléchies » pour aligner les pieds avec l’eau. Ici, la surface de l’eau considérée est la surface non bruitée : cela permet d’éviter des changements trop brusques d’orientation, et de plus cela correspond plus à un comportement physique : une planche de surf passe sous les petites vaguelettes à la surface de l’eau.
* Une image contenant texte

  Description générée automatiquementA ce niveau, la planche (qui n’est liée qu’au pied droit) doit être réalignée : on se sert des positions des pieds avant et arrière pour la recaler.

Remarque : L’angle de roulis n’est pas considéré ici : en effet, dans le cadre d’une planche de surf (cela peut différer pour un véritable kite-surf, où la traction de la voile modifie la physique du phénomène), la planche reste généralement horizontale, car le poids du surfeur est majoritaire devant le moment généré par l’eau sur les bords de la planche.

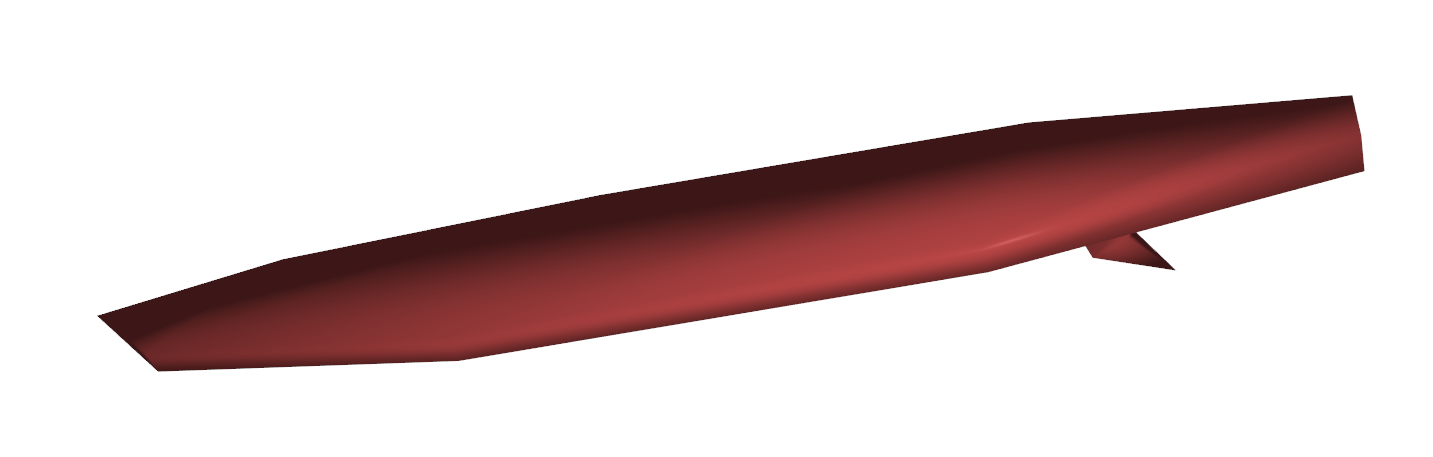
On obtient finalement des positions comme celles-ci (sans surface de l’eau, voir la simulation pour le rendu sur l’eau) :

## Meshes

Une image contenant texte, accessoire, stationnaire, boîtier

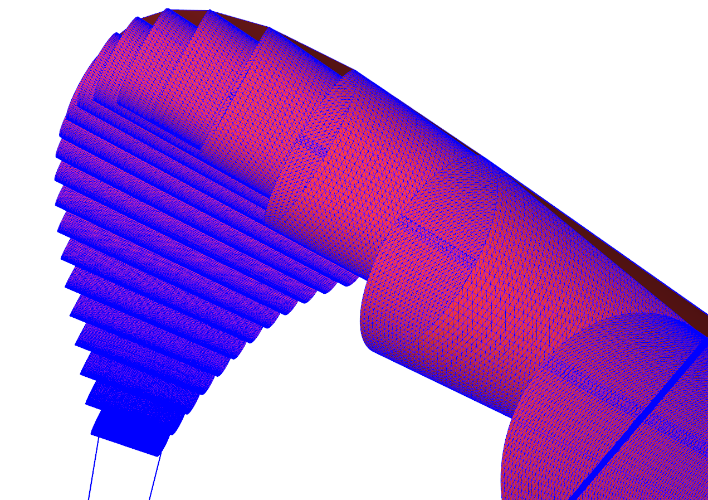
Description générée automatiquementNous avons dû créer manuellement deux maillages pour notre scène. Nous avons volontairement décidé de les dessiner par nous-même plutôt que d’en importer, pour garder une cohérence dans le style de notre scène, et pour avoir une plus grande flexibilité des paramètres. Pour plus de détails, voir *meshes.cpp*

### Planche

Une image contenant stationnaire, stylo, jour

Description générée automatiquement26 Points

### Voile

Assemblage de 21 cylindres ouverts, 20 quadrangles et 4 cylindres pour les cordes.

## Océan

### Forme, paramètres

Concernant la forme, nous avons choisi de générer notre océan sur la base d’une somme de contributions de type onde sphériques atténuée, de vitesse de phase *v* constante :

Dans notre simulation, nous avons 4 points générateurs, situés bien au-dehors de la zone affichée :

Une image contenant texte

Description générée automatiquement

L’océan obtenu ainsi étant entièrement lisse, nous avons décidé d’ajouter des vaguelettes, et de créer un effet d’écume en affichant une deuxième surface d’océan, moins bruitée que la première, et avec des paramètres d’ombrage et couleur différents.

### Ajouts

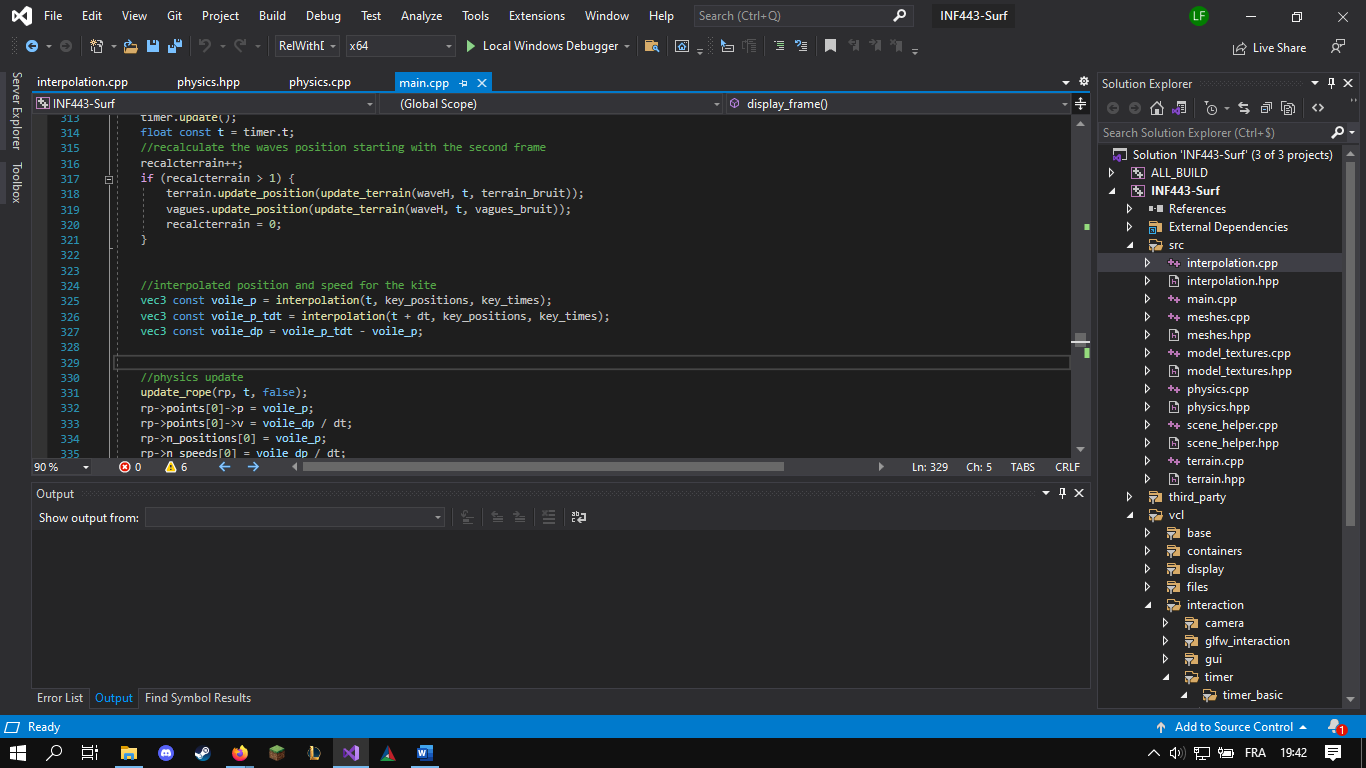
L’ajout de bruit (vaguelettes) se fait au moyen d’un bruit de Perlin 2D, ici additif :

Le bruit se déplace le long de l’axe y, avec une vitesse égale à vit/2 (ici 40 affiché en raison de la conversion en coordonnées de texture – coordonnées globales) pour augmenter le réalisme.

La deuxième surface d’eau est identique à la première, mais moins bruitée et plus claire. Etant donné que ces deux surfaces sont à la même altitude en de nombreux points, il y a un phénomène de Z-fighting (facettes superposées entrant en collision dans le Z-Buffer), ici volontairement conservé car donnant un effet de scintillement à notre surface aquatique.

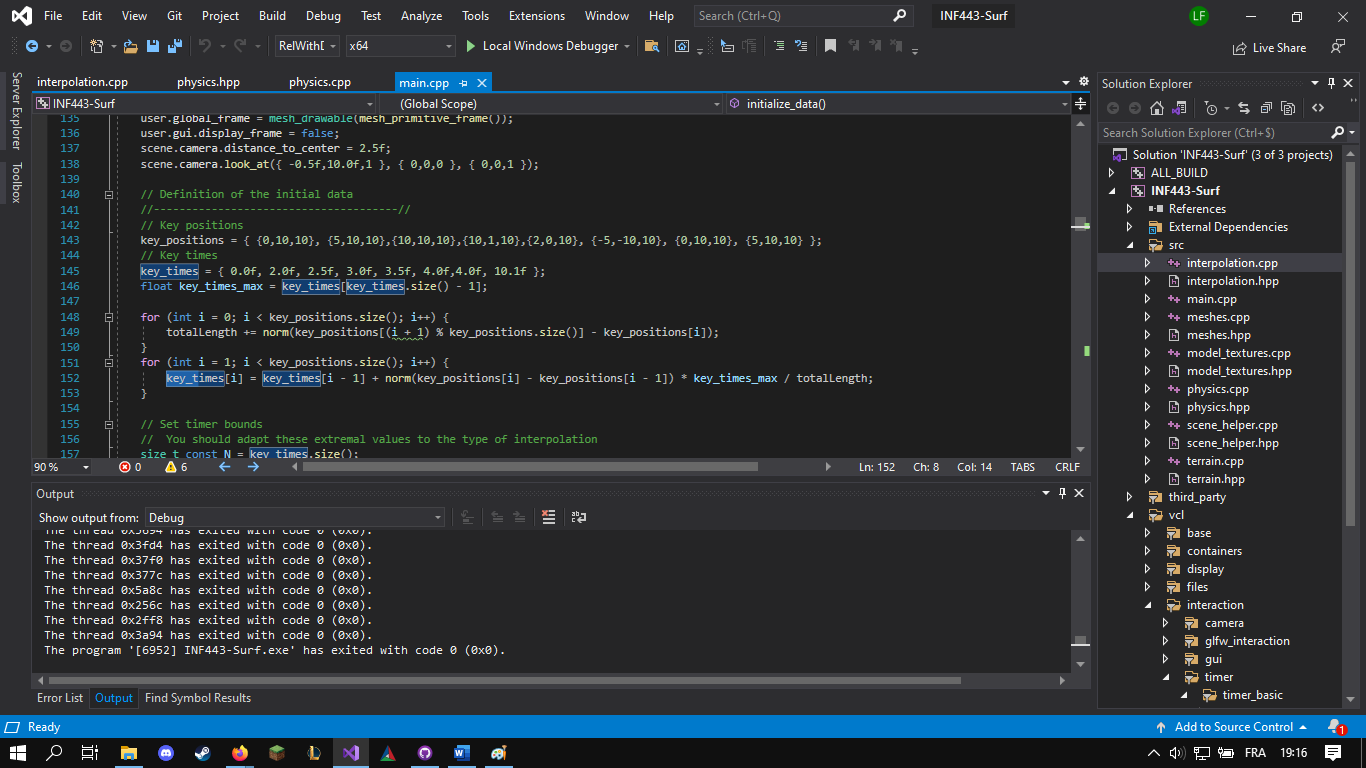
## Simulation physique

### Voile

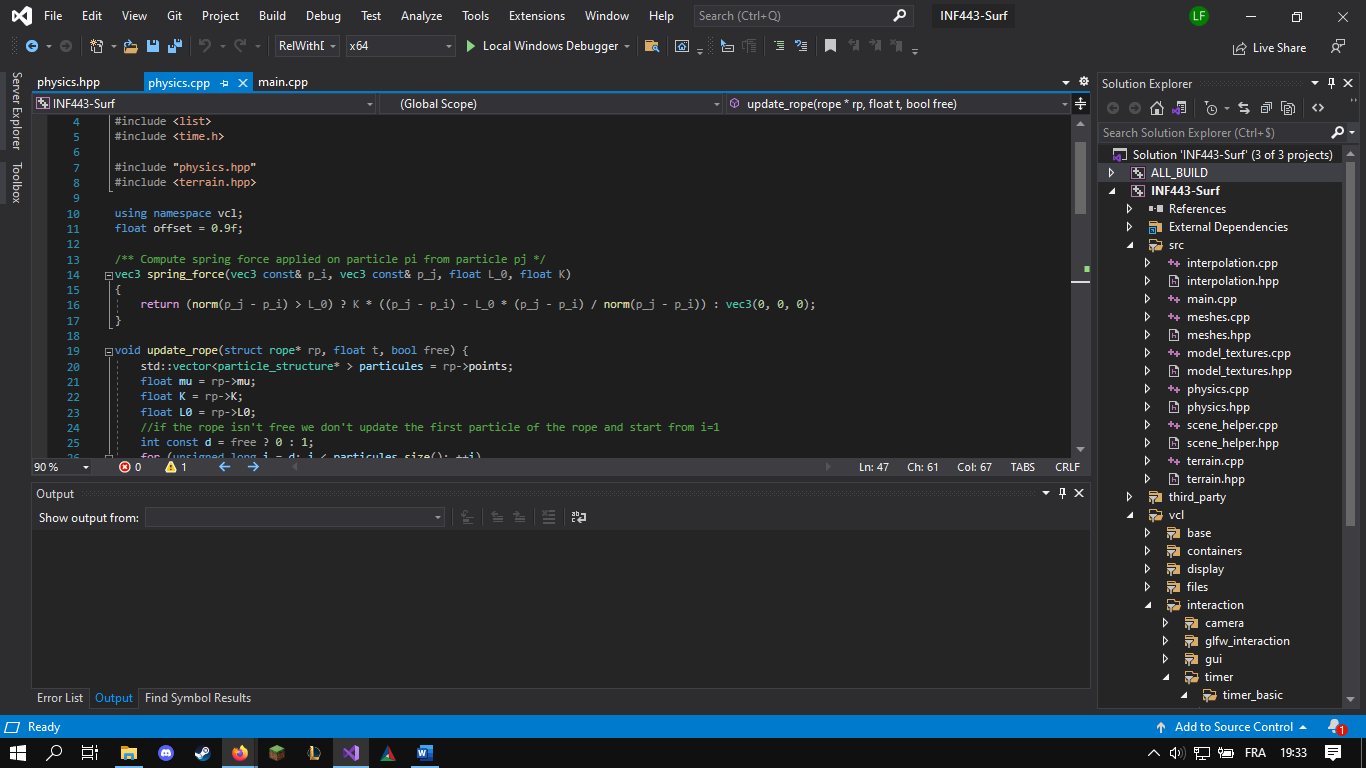
La voile est positionnée par interpolation spline cardinale. A partir de cette interpolation on déduit également la vitesse et l’orientation de la voile en comparant les positions en t et en t+dt.

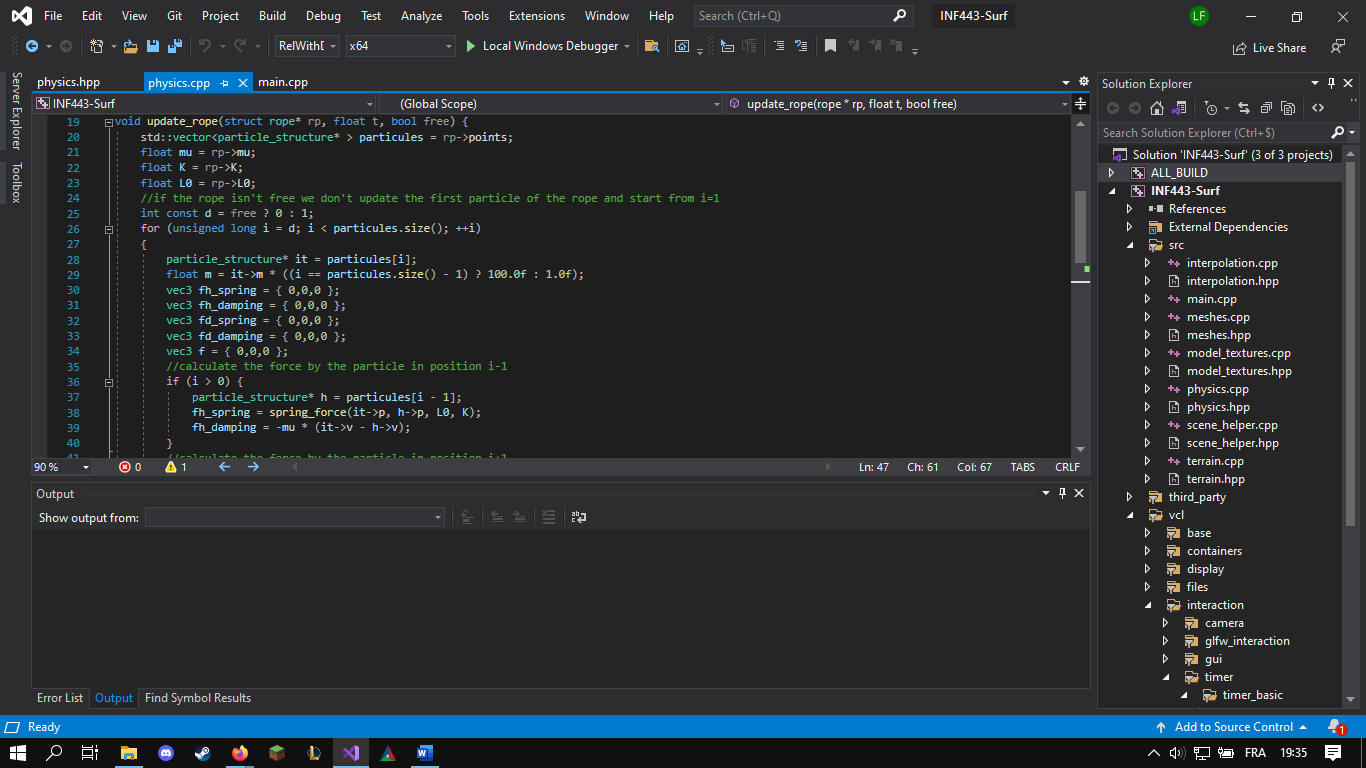
Les positions et les temps clés sont donnés en entrée, mais les temps clés sont mis à jour afin d’avoir une vitesse constante du cerf-volant au cours du temps : seul les temps extrêmes sont conservés.

Pour cela on ajuste l’écart temporel entre deux points pour qu’il soit proportionnel à la distance rapportée à la somme des distances entre points.

Nous avons gardé cette architecture afin de pouvoir aisément revenir à des temps clés fixés, en commentant la boucle de calcul des temps.

### Corde

La corde est animée par simulation physique en la considérant comme une chaîne de ressorts en traction, c’est-à-dire qu’ils n’exercent une force que si leurs longueurs sont supérieures à sa longueur de repos (ils ne poussent pas le surfeur, ils tirent uniquement).

Le dernier élément de la corde est en réalité le surfeur qui y est attaché. Il a donc une masse supérieure à celle du cerf-volant, ici d’un facteur 100.

La simulation d’une corde ainsi modélisée en temps réel nous posant des problèmes de convergence, nous avons finalement décidé de simuler la corde par un seul ressort, mais d’afficher un profil de corde.

A picture containing sport, colorful, line

Description automatically generatedPour ce faire, on crée un deuxième objet de type corde à l’aide des positions du cerf-volant, du surfeur, et d’une parabole.

Enfin la réaction du surfeur au contact de l’eau est une poussée verticale le ramenant à la surface. Cela permet de représenter le fait que la planche empêche le surfeur de couler (poussée d’Archimède), sans passer par des calculs de résistance de l’eau, de tension de surface etc. De plus, la difficulté à simuler une corde « infiniment rigide » sans divergence rendait peu satisfaisant la simulation physique avec une poussée d’Archimède classique.

On obtient ainsi un surfeur, tiré par une voile, qui suit la surface de l’eau.

### Disco !

Pour le plaisir des yeux et de la programmation, nous avons implémenté une petite surprise : cochez la case « Secret » pour le découvrir !



*Texture « unwrapped » utilisée pour la tête*

*Source : APPLICATION OF COMPUTER MODELING FOR PLANNING PLASTIC SURGERIES Klaudia Jamrozik, Jakub Rusek, Dominik Szozda, Krzysztof Karbowski*