Cahier des charges- moteur 3D StarMap

OpenGL-ES pour Android/iOS/Windows

Table des matières

Objectif

Le moteur 3D de la StarMap doit permettre l'intégration d'une représentation en 3D d'une carte du ciel dans les systèmes Android, iOS et Windows.

Le moteur 3D doit permettre une communication dans les 2 sens entre la couche de code native des plateformes cible et sa couche OpenGL.

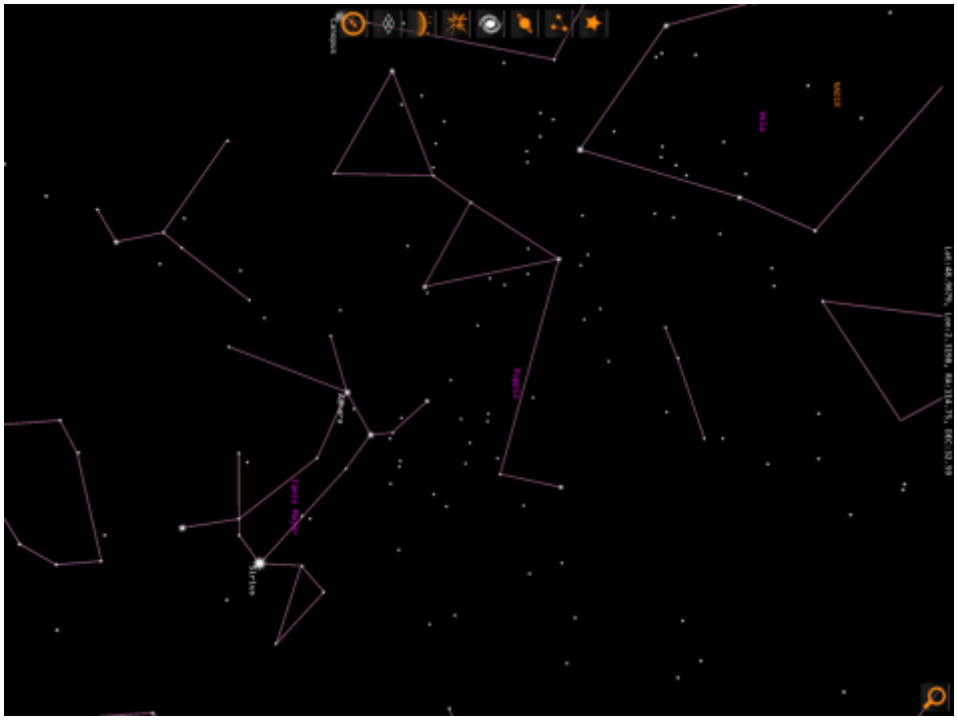
Plateformes ciblées

Les plateformes ciblées par la société Stargazers.space sont à minimum iOS 9, Android 4.4 et Windows 10.

Dans un premier temps, seul les plateformes mobiles sur tablette (mode paysage uniquement) sont visées. Mais à moyen terme nous souhaitons un rendu sur TV (Android TV, Xbox, Apple TV) et sur smartphone.

Rendu cible

À la vue des différentes applications présentes sur les marchés mobiles nous avons délimité notre souhait entre 2 bornes précise (rendu minimum et cible souhaité)



Borne minimal : Application « Sky Map » sur iPad. Modèle de rendu simple avec les constellations représentées uniquement en fil de fer par des traits violets.

Rendu cible : Application « Carte du ciel » sur iPad. Modèle riche avec des effets de lumières variable sur les étoiles ainsi que des images de fond pour représenter les constellations.

Représentation graphique

Généralités

La représentation graphique doit se faire depuis un point central qui est l'utilisateur (sa position géographique ou un point spécifié). Ce point central ne peut pas être modifié une fois le moteur lancé. Pour en changer, il faut repasser dans l'application et en choisir un nouveau avant de relancer le moteur 3D.

En termes de projection, nous souhaitons éviter le plus possible les effets de déformation en bordures d'écran lorsque le FOV est très élevé (projection orthographique ? ou autre ?)

La représentation des éléments à l'écran est donnée pour un instant T. Cependant le facteur temps fait varier la position des objets. Ainsi une étoile au 10 janvier 2017 a 15h ne sera pas à la même position à la date du 4 mars 2017 à 15h. Il faut donc que la représentation graphique évolue en fonction du temps sans avoir besoin de recharger entièrement le modèle.

L'équation permettant un déplacement en fonction du temps sera fournie.

Les étoiles

Les étoiles sont des objets lointains et ne sont pas détaillés. Seuls les éléments de magnitude et de colorimétrie peuvent les faire variés graphiquement.

Pour cette application elles peuvent toutes être considérées comme étant à la même distance.

Le zoom sur une étoile ne permet pas d'en afficher une sphère détaillée. Au mieux celle-ci va varier d'intensité mais pas de taille.

Une étoile peut donc être représentée par un point avec un halo de couleur autour qui varie en luminosité (ceci est une proposition, toute autre suggestion est la bienvenue)

Afin qu'une étoile soit représentée à l'écran, elle doit respecter les contraintes suivantes :

- Etre dans le FOV (champ de vision) de l'écran
- L'étoile doit avoir une magnitude suffisante (la limite sera donnée par le niveau de zoom)

Exemple de représentation en zoom (par rapport au changement de magnitude)

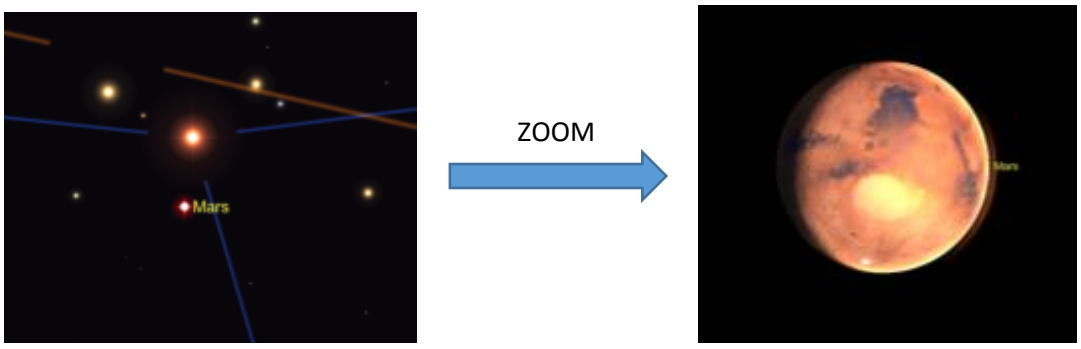
Zoom sur une zone : plus le zoom est effectif, plus on découvre de nouvelles étoiles qui apparaissent (le zoom change le niveau de magnitude minimal nécessaire pour une visualisation d'une étoiles)

Les planètes/lunes/notre soleil

Les éléments proches de nous (les planètes du système solaire, les lunes et notre soleil) sont représentés par des sphères texturées. Ces sphères sont d'un certain rayon lors d'un dé-zoom complet et peuvent être entièrement zoomées pour faire la taille de l'écran pour en voir les détails.

Une subtilité existe pour Saturne qui doit en plus de la sphère, avoir un anneau autour.

Exemple de représentation :



Chaque planète aura une forme identique (sphérique d'un certain rayon)

Les astéroïdes / Comètes

Les éléments annexes comme les comètes et les astéroïdes seront représentés par une forme unique (patatoïdes) texturée en forme de roche. Les comètes auront en plus une représentation d'une queue de comète.

Chaque élément aura une forme identique.

Autres

Nous souhaiterions dans un moyen terme, permettre l'affichage d'éléments autre comme l'ISS ou les satellites.

Nous aimerions afin de mettre ceci en place pouvoir importer dans le moteur 3D un éléments 3D directement construit et lui fournir des coordonnées.

Cette feature est à spécifier selon les possibilités à la produire (en termes de modèle 3D importable, de paramètres nécessaires…)

Interactions possible

Un certain nombre d'interaction doivent être géré par le moteur 3D pour l'expérience utilisateur :

- Le moteur 3D doit permettre un déplacement du ciel (en mode sphérique) autour du point central.
- Le moteur 3D doit permettre de gérer le niveau de zoom afin de pouvoir zoomer sur certain objet et en faire apparaitre des nouveaux (définit par la magnitude)
- Le moteur 3D doit être capable de capturer la zone de clique/touch de l'utilisateur pour sélectionner l'élément le plus proche de cette zone.
  Un calque visuel est à ajouter par-dessus un élément sélectionner pour indiquer à l'utilisateur la réussite de son action. Un second clique/touch permet d'enlever la sélection de l'utilisateur.
- Un mode réalité augmenté qui permet en fonction du positionnement dans l'espace de la tablette, d'orienter la carte du ciel
- Appliquer certains filtres (afficher/masquer certains éléments comme les constellations, les galaxies…)

Les entrées et sortie

Les entrées

Le moteur 3D a besoin d'un certain nombre de paramètre que l'application native doit fournir pour un affichage correct et optimale.

- Une liste d'étoiles à placer selon des coordonnées en radians (en ascension droite et en déclinaison)
- La couleur de chaque étoile
- La magnitude de chaque étoile
- Les planètes avec leur coordonnée et leur texture et leur équation de trajectoire (cf représentation graphique)
- Les astéroïdes et les comètes avec leur coordonnée et leur texture et leur équation de trajectoire (cf représentation graphique)
- Un paramètre date et heure pour adapter la position des objets
- Quelles étoiles peuvent être reliées entre elle et dans quel ordre afin d'en dessiner la constellation
- Des représentations 2D (image png ou autre) à placer derrière les constellations, les galaxies…
- Des points fictifs (non visible sur la carte) pour les relier entre eux afin de dessiner une trajectoires (ligne visible que si l'objet est sélectionné). Ces points peuvent être fournit au cas par cas lors d'une sélection.
- Des modèles 3D pré construit avec leur coordonnée (ISS, Hubble…)
- Une direction de visée pour un éventuelle mode en réalité augmenté (avec positionnement de la tablette dans l'espace)
- Les filtres on/off à appliquer pour afficher/masquer des éléments (changement possible à la volée sans avoir à recharger le moteur)
- Sélectionné ou désélectionné un objet par son identifiant. Par exemple l'utilisateur demande à voir Mars depuis un overlay, la carte se déplace automatiquement sur l'objet Mars qui sera sélectionné.

Les sorties

Le moteur 3D doit permettre de fournir au code natif certains éléments.

- Fournir l'identifiant de l'objet le plus proche de la zone de clique/touch de l'utilisateur. Cela pour permettre d'afficher un overlay en code natif donnant des précisions sur l'objet sélectionnée.
- Indiquer qu'un objet a été désélectionné. Cela pour permettre d'enlever l'overlay.

Communication entre OpenGL et code natif

La communication entre la couche OpenGL du moteur 3D et le code natif peut se faire via de simple méthode à appeler avec des paramètres.

Le catalogues des éléments peut être sous forme d'un tableau de data pré formaté selon le besoin. Un csv, un Json, un tableau de donnée… sont des solutions envisageables en fonction du besoin coté openGL-ES.

Temps de développements

Fonction	Estimation initiale du temps	Temps effectifs	Modifié par
Total	15 jours		Stargazers.Space