|  |
| --- |
| [Type the company name] |
| [Type the document title] |
| [Type the document subtitle] |
|  |
| **KaL** |
| **[Pick the date]** |

|  |
| --- |
| [Type the abstract of the document here. The abstract is typically a short summary of the contents of the document. Type the abstract of the document here. The abstract is typically a short summary of the contents of the document.] |

Table des matieres

[Remerciements 3](#_Toc232965650)

[Introduction 4](#_Toc232965651)

[Presentation de nVidia CUDA 5](#_Toc232965652)

[Implementation des algorithmes 5](#_Toc232965653)

[Architecture de notre environnement de travail 5](#_Toc232965654)

[Strategie de parallelisation 5](#_Toc232965655)

[Frustum Culling 5](#_Toc232965656)

[Frsutum Pyramidal / Axis-Aligned Bounding Box 5](#_Toc232965657)

[Frustum Pyramidal / Sphere 5](#_Toc232965658)

[Frustum Spherique / Axis-Aligned Bounding Box 5](#_Toc232965659)

[Frustum Spherique / Sphere 5](#_Toc232965660)

[Occlusion Culling 5](#_Toc232965661)

[Methode utilisee 5](#_Toc232965662)

[Occlusion dans un Frustum Pyramidal 5](#_Toc232965663)

[Occlusion dans d’autres Frustums 5](#_Toc232965664)

[Resultats 5](#_Toc232965665)

[Approfondissements 5](#_Toc232965666)

[Structure de donnees pour l’acceleration 5](#_Toc232965667)

[Construction de la structure 5](#_Toc232965668)

[Lecture de la structure 5](#_Toc232965669)

[Conclusion 5](#_Toc232965670)

[Works Cited 6](#_Toc232965671)

# Remerciements

Nous tenons a remercier chaleureusement …

# Introduction

Dans le cadre de l’UV TX52 (Travaux de Laboratoire) proposee a l’UTBM, nous avons etudie des algorithmes de calcul de visibilite comme le Frustum Culling ou l’Occlusion Culling.

L’objet de nos travaux était d’evaluer la parallelisation potentielle de ces algorithmes, afin de les adapter pour etre executes sur des cartes graphiques programmables.

La possibilite d’executer ces algorithmes de manière parallele permettrait de les utiliser afin, par exemple, d’evaluer la visibilite d’entites dans les champs de vision d’agents autonomes au sein d’une plateforme de simulation et/ou de realite virtuelle.

Ces algorithmes proviennent du monde de l’imagerie numerique, et generalement, un seul calcul par rafraichissement d’image s’avere necessaire, pour les applications considerees. Dans le cas d’une application de simulation avec des agents autonomes possedant un champs de vision, ces algorithmes doivent donc s’executer plusieurs fois a chaque rafraichissement de l’univers. Notre tache est donc d’evaluer la performance de ces algorithmes utilises en parallele sur des processeurs graphiques programmables.

Pour implementer ces algorithmes, nous avons utilise la technologie nVidia CUDA, qui permet de programmer directement en C des processeurs graphiques.

Dans une premiere partie, nous presenterons la technologie utilisee, ses specifites, ses avantages et ses contraintes.

Ensuite nous etudierons les algorithmes parallelises, ainsi que leurs implementations adaptees pour CUDA.

Nous presenterons egalement les resultats que nous avons obtenus en utilisant ces algorithmes, et les comparerons avec l’existant.

Puis nous verrons quelles ameliorations il serait possible d’apporter a nos travaux, nottament l’utilisation eventuelle de structure de partitionnement de l’espace.

# Presentation de nVidia CUDA

# Implementation des algorithmes

## Architecture de notre environnement de travail

## Strategie de parallelisation

## Frustum Culling

### Frsutum Pyramidal / Axis-Aligned Bounding Box

### Frustum Pyramidal / Sphere

### Frustum Spherique / Axis-Aligned Bounding Box

### Frustum Spherique / Sphere

## Occlusion Culling

### Methode utilisee

### Occlusion dans un Frustum Pyramidal

### Occlusion dans d’autres Frustums

# Resultats

# Approfondissements

## Structure de donnees pour l’acceleration

### Construction de la structure

### Lecture de la structure

# Conclusion

# Works Cited