HAL
open science

Modélisation physiquement réaliste de simulation d'entraînement maritime

Jean-Marc Cieutat

To cite this version:

Jean-Marc Cieutat. Modélisation physiquement réaliste de simulation d'entraînement maritime. Synthèse d'image et réalité virtuelle [cs.GR]. université de bordeaux 1, 2003. Français. NNT : . tel-01707710

HAL Id: tel-01707710

https://hal.science/tel-01707710

Submitted on 13 Feb 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

N° d'ordre : 2771

Thèse

Présentée à

L'Université de Bordeaux I

École doctorale de Mathématiques et d'Informatique

Par **Jean-Marc CIEUTAT**

Pour obtenir le grade de

Docteur

Spécialité : Informatique

Modélisation physiquement réaliste de session de simulation d'entraînement maritime

---

Soutenue le : 17 décembre 2003

Après avis des rapporteurs :

Fabrice Neyret ..... rapporteur
Bruno Arnaldi ..... rapporteur

Devant la commission d'examen composée de :

Pascal Guitton (Professeur, Université Bordeaux 1)
Fabrice Neyret (Chargé de recherche CNRS, GRAVIR/IMAGIS)
Bruno Arnaldi (Professeur, INSA Rennes)
Michel Nakhlé (Docteur d'état, consultant Compagnie des Signaux)
Nadine Rouillon-Couture (Chargée d'enseignement et de recherche, LIPSI/ESTIA)
Christophe Schlick (Professeur, Université Bordeaux 1)

Rapporteur de soutenance :

Pascal Guitton (Professeur, Université Bordeaux 1)

Table des matières

Introduction …… 8

1 Le cadre de nos travaux ……11

1.1 Introduction …… 11
1.2 Notre simulateur d'entraînement maritime …… 11
1.3 Le marché des simulateurs d'entraînement maritime …… 13
1.4 Les autres domaines d'application …… 15

2 Un peu d'océanographie ……16

2.1 Introduction …… 16
2.2 Typologie des vagues océaniques …… 16
2.3 Le processus de génération et de propagation des vagues océaniques ……17
2.4 Les courants marins ……20

3 Théories physiques et modélisation des vagues océaniques ……23

3.1 Introduction …… 23
3.2 Théories physiques …… 23
3.2.1 Les équations de la mécanique des fluides …… 23
3.2.2 La forme linéaire des équations de Navier-Stokes ……24
3.2.2.1 Réécriture des équations de Navier-Stokes ……24
3.2.2.2 Vagues d'amplitude finie ……25
3.2.2.3 Vagues de faible amplitude …… 26
3.2.3 La théorie de Gerstner ……29
3.3 Modélisation des vagues en infographie …… 32
3.3.1 Simulation complète du fluide …… 32
3.3.1.1 Le modèle de Foster …… 32
3.3.1.2 Le modèle de Chen et Lobo …… 33
3.3.1.2 Le modèle de Layton …… 33
3.3.2 Simulation d'une onde de surface ……34
3.3.2.1 Le modèle de Fournier et Reeves ……34
3.3.2.2 Le modèle de Gonzato …… 35
3.3.2.3 L'algorithme de lancer de vague ……36
3.3.3 Simulation spectrale …… 38
3.3.3.1 Le modèle de Mastin …… 38
3.3.3.2 Le modèle de Tessendorf …… 38

4 Extensions et apports sur la modélisation des vagues océaniques ……42

4.1 Introduction …… 42
4.2 Simulation d'une onde de surface …… 42
4.2.1 Interactions vague-courant ……42

4.2.2 L'algorithme de lancer de vague à partir
d'un arc de cercle ……47
4.3 Somme classique des vagues et simulation spectrale …… 49
4.3.1 La somme classique des vagues …… 49
4.3.2 Extensions du modèle de Tessendorf …… 51
4.4 Vagues de courtes longueurs d'onde ……56
4.5 Résumé du chapitre …… 57

5 Simulation temps réel des océans ……59

5.1 Introduction …… 59
5.2 Enoncé des contraintes ……60
5.3 Notre structure de données …… 61
5.4 Pré-traitements des données initiales ……63
5.4.1 Le littoral et la mer ……63
5.4.2 La profondeur et la pente du fond de la mer ……64
5.4.3 Les vecteurs vents et courants de marée ……66
5.5 Architecture logicielle ……67
5.6 Exclusion du cône de vision ……69
5.6.1 Principe général ……69
5.6.2 Exclusion au niveau des tuiles ……70
5.6.3 Exclusion au niveau des grilles MNT …… 71
5.7 Affichage par niveaux continus de détails des surfaces de mer ……72
5.7.1 Travaux précédents ……72
5.7.1.1 Subdivision hiérarchique à l'aide d'arbre quaternaire ……73
5.7.1.2 Subdivision hiérarchique à l'aide d'arbre binaire ……74
5.7.1.3 Réseaux irréguliers de triangles isocèles à angles-droit ……75
5.7.2 Notre méthode d'affichage de niveaux continus de détails ……76
5.8 Résumé du chapitre …… 82

6 Éclairage temps réel des océans …… 84

6.1 Introduction …… 84
6.2 Travaux précédents ……84
6.3 Notre modèle d'éclairage temps réel des océans …… 85
6.4 La programmation par « shader » ……91
6.5 Les textures et la technique du « bump mapping » ……95
6.6 L'écume …… 97
6.7 Résumé du chapitre …… 98

7 L'effet des vagues sur le comportement du navire ……100

7.1 Introduction …… 100
7.2 Le calcul de la pression sur la surface mouillée de la coque …… 100

Conclusion ……103

Annexe A : Dossier technique du simulateur installé au lycée de Ciboure 106
Annexe B : Glossaire des termes marins …….....…… 116
Annexe C : “State of the art of maritime trainning simulators” …….….....……118
Annexe D : Complément sur l’utilisation de la Transformée de Fourier …….....……122
Annexe E : Un exemple de vertex shader et de pixel shader …….....… 128

Bibliographie …….....…130

Table des figures

1 Première version du simulateur d'entraînement maritime ...…….. 12
2 Un simulateur de type « maquette échelle 1 » 14
3 Définition d'une vague …….16
4 Exemple de caractéristiques des vagues en fonction
du vent générateur …….....…… 19
5 Houle opposée au courant des Aiguilles au large de la côte Sud-Africaine[Ifrem] 20
6 Les courants océaniques de surface ……..21
7 Carte des courants de marées [IM] …….....……...22
8 Mouvements d'un objet à la surface de l'océan [Ifrem] ……..……...28
9 Vagues de faible amplitude ……..28
10 Les vagues de Gerstner …… 30
11 Propagation des vagues de Gerstner …….……..30
12 Vue de la surface de la mer en fonction de k et de R ……...31
13 Équations en eaux peu profondes 33
14 Théorie de Biesel …….……..34
15 Le phénomène de réfraction …….……37
16 L'algorithme du lancer de vague …..…… 37
17 Propagation en présence d'un courant contraire de plus en plus fort .....…….... 45
18 Propagation du front de vague …… 48
19 Définition du front initial de vague …..…… 48
20 Etats de mer (4 à gauche), (5 à droite) …….....……..……52
21 Somme de vagues de Biesel avec les FFT …….…….. 54
22 Notre modèle de mer du vent ….....……. 54
23 Étapes successives pour calculer le nouveau champ d'amplitudes ….....…….……..56
24 Les vagues de courtes longueurs d'onde ……....…….…..57
25 Surfaces de mer avec un point tous les 100 cm, 10 cm, 1 cm …….…….60
26 Images de synthèse du film Titanic ….…….60
27 Définition du cône de vision ….. ……..61
28 Découpage du paysage en tuiles …….. 62
29 Découpage récursif d'un niveau ……...63
30 Points d'altitude nulle n'appartenant pas à la mer ……..……64
31 Superposition des surfaces de la mer et de la côte ……....…..…..…….. 64
32 Le Golfe de Gascogne ……...……65
33 Transformation d'une ligne bathymétrique ……...……65
34 Ligne bathymétrique locale ……..……66
35 Interpolation bilinéaire …….…… 66
36 Le logiciel ARPEGE …… 67
37 Notre architecture logicielle …….68
38 Parallélépipède englobant et cône de vision …… 70
39 Inclusion des navires dans le cône de vision …….…...71
40 Fermeture des surfaces de la mer et de la côte avec les plans du cône de vision ...….…71

41 Triangulation d'un arbre quaternaire ……..……. 73
42 Projection du delta-segment ……..……74
43 Niveaux 0 à 5 d'un « bintree » ……..…… 74
44 Propagation récursive des subdivisons ……..……75
45 « bintree » et coordonnées des sommets des triangles ……76
46 Une suite de triangles adjacents ……..……78
47 Etude comparative des structures de données de types « bintree » et grille régulière …. 78
48 Densité des réseaux de triangles …….. 79
49 Etude comparative des suites de triangles et des suites de quadrilatères adjacents …… 79
50 Craquelures entre les maillages ……..…..80
51 Plan texturé sous la surface de la mer ……..…80
52 Diminution du nombre de points ……..…..81
53 Intervalles d'erreur ……... 82
54 Les vecteurs réflexion et réfraction ……..86
55 Contrastes entre les crêtes et les creux des vagues ……...87
56 Approximation des coefficients de Fresnel …….. 87
57 Les reflets sur la mer et la visibilité du fond ……88
58 Texture hémisphérique du ciel ……..… 89
59 Couleur de l'eau en fonction de l'angle de réfraction ……..…91
60 La programmation par « shader » [ATI] ….....…… 92
61 Etude comparative du rendu effectué par le CPU ou la GPU ……...… 93
62 Paramètres des programmes d'éclairage ….....……..……94
63 La réflexion du soleil dans un « vertex shader » ou dans un « pixel shader » ….……95
64 Algorithme de « bump-mapping » …….....……..……..96
65 Les vecteurs tangentes à la surface de la mer 96
66 L'ajout de l'écume [Tes01-2] .....…… 97
67 Les degrés de liberté du navire [Ped00] ……..…… 101
68 Courbe de stabilité transversale d'un navire ……102
69 La salle de classe : le poste enseignant et les passerelles 106
70 Préparation d'un exercice de pointage radar 107
71 Préparation d'un questionaire à choix multiple 108
72 Gestion de la base de données des élèves 108
73 Supervision d'une session de simulation 109
74 Ensemble de deux passerelles 110
75 Homme de barre (au commerce) / Homme de quart ou patron (pêche) 110
76 Officier de quart ou commandant (au commerce) / Patron (pêche) 111
77 Le port de Saint-Jean de Luz 111
78 Vue de la passerelle 112
79 La pêche à la senne 112
80 Le radar 112
81 Le sonar 113
82 Le sondeur 113
83 L'interface de signalisation 114

Introduction

En 1998, l'École Supérieure des Technologies Industrielles Avancées [Estia], situé géographiquement à Biarritz, a développé un simulateur d'entraînement maritime qui a été installé au lycée maritime de Ciboure. À cette époque, faute de ressources de calcul et d'affichage, il n'était guère possible de proposer une représentation réaliste de ce qu'il est communément appelé, en termes marins, les *états de mer*. Comme aujourd'hui nous disposons de plus de ressources de calcul et d'affichage qu'en 1998, les objectifs de nos recherches sont la modélisation et le rendu en temps réel des vagues océaniques ainsi que l'étude mécanique de leurs incidences sur le comportement d'un navire. Pour y parvenir, nous nous baserons sur l'étude phénoménologique des effets du vent, des courants marins et de la profondeur sur les vagues océaniques.

Les enjeux de cette étude sont importants. Il existe malheureusement plus d'accidents dans les métiers de la mer que dans beaucoup d'autres métiers. Dans ce contexte, la simulation d'entraînement sur ordinateur prend tout son sens. Il existe même une résolution de l'Organisation Maritime Internationale (OMI) obligeant tout personnel naviguant sur une passerelle à suivre un stage de simulation une fois l'an. Toutefois, il faut bien reconnaître que cette résolution est peu appliquée de par la difficulté technique que cela représente. La plupart des accidents maritimes se produisent dans des conditions de mer agitée difficiles à reproduire sur ordinateur. Les situations qui conduisent à l'accident sont nombreuses : les hauteurs des vagues étaient telles que la visibilité était réduite et que la collision n'a pas pu être évitée ; le navire n'a pas pu maintenir un cap perpendiculaire au front des vagues et il a chaviré ; le courant et les vagues ont dévié le navire de sa route qui s'est échoué ; lorsque le chalutier remontait son chalut, la tension s'exerçait sur l'un des deux câbles et le chalutier a chaviré lorsqu'il a pris une vague de côté ; ...

Tout d'abord, nous décrivons plus précisément nos objectifs de recherche dans le premier chapitre. Ils se traduisent par la réalisation d'un nouveau cahier des charges, que le lecteur pourra comparer avec le dossier technique du simulateur installé en 1998 qui est présenté en *annexe A*. Nous rapportons également dans ce chapitre les résultats d'une enquête menée auprès des principaux fabricants européens de simulateurs d'entraînement maritime. Nous désirions procéder préalablement à un état de l'art de ces simulateurs et connaître l'opinion des industriels sur nos objectifs de recherche. Nous ne pouvions pas aborder ce chapitre sans employer les termes usuels propres au domaine d'application. Pour une meilleure compréhension, nous invitons le lecteur à se reporter en *annexe B* où un glossaire des termes marins est présenté. Enfin, nous ne pouvions pas terminer ce chapitre sans faire référence aux travaux portant sur le rendu de paysages exploités dans d'autres domaines d'application comme les jeux vidéos.

Puis nous réunissons dans le chapitre 2 les principales observations océanographiques liées au processus de génération et de propagation des vagues océaniques dont nous disposons. Ce processus très complexe, auquel se rapportent de nombreuses théories, est étudié par les océanographes depuis plusieurs dizaines d'années.

Le chapitre 3 porte sur la physique de la vague. En infographie, la communauté scientifique se réfère à 2 ou 3 approches seulement. L'approche la plus rigoureuse repose sur une simulation complète du fluide à partir des équations de Navier-Stokes mais c'est aussi l'approche qui nécessite le plus de ressources de calculs. À l'échelle de l'océan, une simulation complète du fluide en temps réel est difficilement envisageable. Les deux autres approches sont la simulation d'une onde de surface et la simulation spectrale. La première d'entre elles est généralement utilisée pour simuler la forme et la propagation d'une onde de surface près des côtes. Dans ce cas, nous sommes en présence d'une **houle** très régulière comme elle est constituée d'un seul train de vagues. La deuxième d'entre elles permet de simuler en pleine mer la génération de plusieurs vagues sous l'effet du vent. Dans ce cas, il s'agit de la **mer du vent** pour laquelle, à des fins de simplification, l'aire génératrice est supposée être éloignée des côtes.

Pour représenter le processus complet de génération et de propagation des vagues océaniques, il est nécessaire de simuler la forme et la propagation de plusieurs ondes de surface depuis la pleine mer jusqu'au bord du rivage. Une manière d'y parvenir est de pré-calculer, conformément à la première des deux approches citées précédemment, la forme et la propagation de chaque onde de surface à l'aide de l'algorithme connu sous le nom de « l'algorithme de lancer de vague ». Au cours des pré-calculs, la forme et la propagation de chacune des ondes de surface sont mémorisées au sein de grilles 2D qui recouvrent l'ensemble de la zone de navigation. Lors d'une session de simulation, chaque onde de surface est ensuite reconstituée en 3D avant que la somme ne soit effectuée. On peut ainsi prendre en considération les effets simultanés du vent, du courant et de la profondeur. Toutefois, l'exploitation temps réel de cette approche est conditionnelle à une augmentation toujours croissante de la puissance de calcul. De plus, toute augmentation des ressources de calcul peut être consommée en augmentant le nombre de vagues à traiter.

Aussi, dans le chapitre 4, nous limitons notre champ d'investigations à un modèle de houle régulière, plus complet que dans le passé, qui permet, en temps réel, de simuler la forme et la propagation d'une seule onde de surface depuis la pleine mer jusqu'au bord du rivage. Lorsque plusieurs trains de vagues sont en présence, nous proposons une extension de l'approche de simulation spectrale. À une étude seule des corrélations existantes entre des conditions de vent et un spectre de vagues, nous souhaitons également ajouter les phénomènes les plus visuels dus au courant de marée et à la profondeur. Un complément sur l'utilisation de la transformée de Fourier est présenté en *annexe C*. Au final, un enseignant dans le domaine maritime pourra ainsi reproduire diverses configurations à l'aide des seuls paramètres d'entrée du système : simulation d'une onde de surface ou simulation spectrale, fonds marins, profondeurs, force et direction du courant de marrée, force et direction du vent.

Afin qu'un programmeur puisse facilement implémenter nos modèles de vagues océaniques, nous détaillons dans le chapitre 5 une architecture logicielle adaptée à cette problématique. Beaucoup de simplifications sont proposées dans ce chapitre pour satisfaire la contrainte de temps réel. De manière générale, ces simplifications ont pour effet de diminuer le nombre d'éléments de la scène 3D sans que, pour autant, l'œil de l'observateur perçoive une différence.

Le chapitre 6 est consacré à l'éclairage des scènes océanes. Dans un premier temps, nous définissons notre modèle d'éclairage. Puis, comme les fonctions traditionnelles d'éclairage sont inadaptées à notre cas, notre modèle d'éclairage est implémenté au moyen de programmes plus connus sous les noms de « vertex shader » et de « pixel shader ». Nous expérimentons alors les possibilités offertes par les cartes graphiques programmables ; le lecteur trouvera en *annexe D* une illustration de nos expérimentations.

Les effets des vagues océaniques sur le comportement du navire sont seulement abordés dans le dernier chapitre. Il existe principalement deux grandes approches aujourd'hui référencées par la communauté scientifique en mécanique du navire et beaucoup de méthodes s'y rapportant. Nous décrivons, ici, une seule de ces méthodes parce qu'elle donne des résultats acceptables en temps réel et parce qu'elle s'appuie sur la forme linéaire des équations de Navier-Stokes. Elle est plus connue sous le nom de méthode par tranches « strip theory » ou sous l'acronyme STF du nom de ses inventeurs.

Chapitre 1

Le cadre de nos travaux

1.1 Introduction

Nous décrivons dans ce chapitre le cadre de nos travaux. Nous commençons par présenter le simulateur d'entraînement maritime qui a été installé au lycée maritime de Ciboure en 1998. À partir d'une analyse critique de l'existant, nous établissons un nouveau cahier des charges. Pour confronter les nouveaux objectifs fixés avec les préoccupations des industriels, nous leur avons adressé un questionnaire dont nous dépouillons sommairement, dans le deuxième paragraphe, les quelques réponses qui nous ont été retournées. Enfin, pour terminer ce chapitre, nous faisons référence aux domaines d'application connexes avec la simulation d'entraînement.

1.2 Notre simulateur d'entraînement maritime

À la demande du lycée maritime de Ciboure, l'École Supérieure des Technologies Avancées [Estia] a tout d'abord étudié la faisabilité d'un simulateur d'entraînement adapté aux besoins de la formation initiale de l'enseignement maritime. L'obtention d'un financement ANVAR en 1995 lui a permis de constituer une équipe projet, composée de quatre personnes et d'un chef de projet, qui, pendant trois ans, a réalisé une première version de l'outil. Cet outil, qui a été écrit en langage de programmation C++ et en gl, est disponible sur station de travail Silicon Graphics [SGI]. L'installation sur site a été effectuée à la fin de l'année 1998. Le dossier technique du simulateur, qui a été rédigé au début de l'année 1999, est présenté en *annexe A*.

Cet outil est avant tout un outil pédagogique. Il a tout d'abord été pensé pour qu'un enseignant dispose de toutes les fonctionnalités lui permettant de superviser simultanément l'enseignement de 12 élèves en formation. Depuis son poste (fig 1, en haut à gauche), l'enseignant dispose ainsi d'une vue synoptique de chaque poste élève. Il peut, à son gré, arrêter une session de simulation, changer les paramètres de l'exercice, déclencher une avarie et bien d'autres possibilités.

L'apprentissage de chaque élève est personnalisé et progressif. L'historique de ses leçons est sauvegardé. Il commence par apprendre le code maritime et l'appliquer durant les premiers exercices de simulation. Les exercices suivants portent sur les instruments de navigation et de pêche que l'élève apprend à manipuler. Il effectue ensuite ses premières missions de navigation (sauvetage d'un homme en mer ; départ d'un point géodésique pour suivre une route et entrer dans un port (fig. 1, en bas à gauche ; ...). En dernier lieu, l'élève s'exerce aux techniques de la pêche industrielle (capture d'un banc de poissons au chalut de fond et au chalut pélagique ; capture d'un banc de poissons à la pêche à la senne (fig. 1, en bas à droite)). La simulation radar est supportée et conforme aux normes STCW [STCW95] en vigueur (fig. 1, en haut à droite).

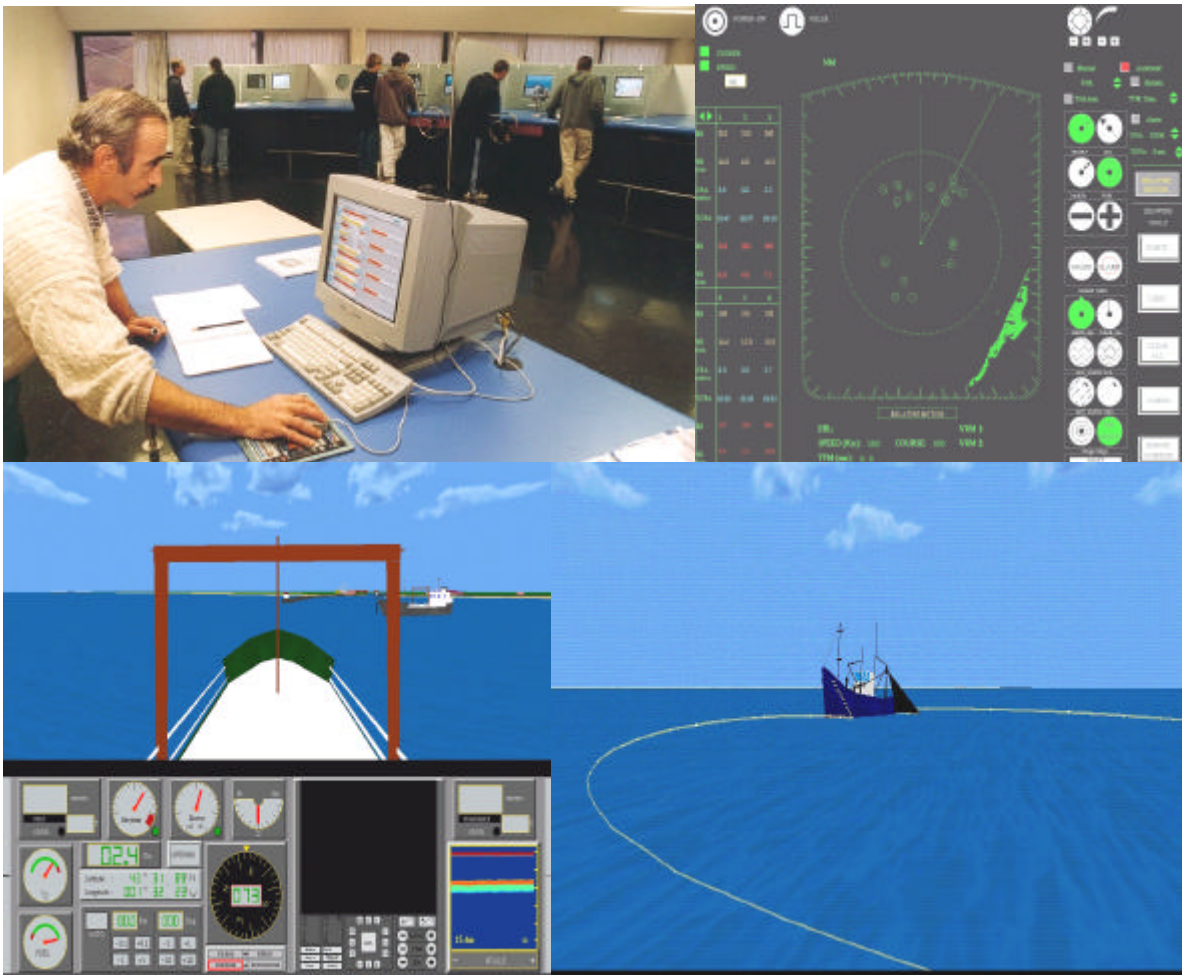


Fig. 1 : Première version du simulateur d'entraînement maritime

Un nouveau cahier des charges

La formation continue est également une des missions du lycée maritime de Ciboure. Dans cette optique, l'établissement souhaiterait disposer de son simulateur pour assurer des prestations de formation continue à des professionnels de la navigation et de la pêche.

Face à des personnes expérimentées, le simulateur risque de manquer cruellement de réalisme. Pour pallier ce manque, une première analyse critique de l'existant devrait permettre de dégager une liste non exhaustive des propriétés désirables de l'outil.

Dans sa version actuelle, les données altimétriques et les données bathymétriques sont très peu exploitées. À titre d'illustration, les paysages côtiers ne sont pas construits à partir de données d'élévation ; ils sont essentiellement « dessinés », ce qui peut amener à constater des différences de position dès lors que la progression du navire est suivie sur une station de cartographie.

Les données relatives aux courants de marée et au vent ne sont également pas exploitées. Quel que soit le lieu, la marée et le vent sont chacun représentés par une unique direction et une unique magnitude ; ils n'ont pas encore été introduits comme étant des champs de vecteurs. Également à titre

d'illustration, les marins exploitent quotidiennement ces données pour optimiser leur déplacement d'un point géodésique à un autre.

Lorsque l'*état de mer* est de zéro sur l'échelle de Beaufort (*annexe B)*, la mer est représentée par un plan texturé (fig. 1, en bas à droite et à gauche). Pour les autres valeurs de l'échelle de Beaufort, la surface de l'océan est modélisée par une fonction sinusoïdale dont l'amplitude varie en fonction de la valeur de l'*état de mer*. Les effets des vagues sur le comportement du navire sont pris en compte au moyen d'une modélisation géométrique. Pour un navire de faible tonnage, les angles d'inclinaison longitudinal et transversal du navire sont directement calculés à partir de la pente de la vague la plus proche de son centre de gravité [Cie01]. Ces deux derniers points nous semblent être les plus critiquables par des marins expérimentés, habitués à traverser les mers quelles que soient les conditions de navigation. Aussi, dès la fin de l'année 1999, nous avons entamé des travaux de recherche sur la modélisation physiquement réaliste de session de simulation d'entraînement maritime.

Les critères que nous avons retenu pour mener nos travaux de recherche

Pour atteindre les objectifs cités précédemment, des améliorations immédiates sont possibles. Par contre, pour obtenir un rendu physiquement plus réaliste, nous nous heurterons à la contrainte de temps réel. **La problématique du temps réel** est de maintenir un taux d'images à peu près constant pendant la simulation et un intervalle de temps constant entre deux images de sorte que l'animation reste fluide. Il en résulte que, à chaque pas de simulation, les calculs que nous pouvons effectuer et le nombre de polygones que nous pouvons afficher dépendent de la puissance du processeur de calcul et de celle de l'accélérateur graphique.

Mais la contrainte de temps réel ne doit pas devenir pour autant obsessionnelle. Selon Gordon Moore [Intel], le co-fondateur de la compagnie Intel, la puissance de calcul des microprocesseurs doit doubler tous les 18 mois. C'est une donnée essentielle qui guide nos choix en vue d'applications industrielles dans un avenir proche.

1.3 Le marché des simulateurs d'entraînement maritime

Nous avons adressé l'enquête de l'*annexe C* aux principaux fabricants européens de simulateurs d'entraînement maritime.

Les deux ténors du marché européen, que sont la compagnie norvégienne KONGSBERG [Kongs] et la société russe TRANSAS [Trans], n'ont pas souhaité nous répondre, conférant à nos réponses un manque de représentativité.

. Depuis 25 ans, la division simulateur de la compagnie norvégienne KONGSBERG a installé environ 500 simulateurs à travers le monde. En France, ce simulateur est installé à l'École Nationale de la Marine Marchande (ENMM) de Saint-Malo [EnmmS] et à celle de Nantes [EnmmN]. Son prix

avoisine le million d'euros. Ce simulateur appartient à la catégorie des simulateurs de type « maquette échelle 1 » (fig. 2) qui visent à reproduire à l'identique une passerelle de navire.

. La société TRANSAS, qui est plus récente, a pu se tailler une part de marché croissante grâce à une politique de prix plus avantageuse que celle de la compagnie KONGSBERG. Le simulateur de navigation TRANSAS de type « maquette échelle 1 » est installé à l'ENMM de Marseille [EnmmM]. La gamme des simulateurs TRANSAS est la plus complète qui existe sur le marché. Entre le simulateur de type «maquette échelle 1» et la configuration d'entrée de gamme constituée d'un seul ordinateur, on trouve une configuration intermédiaire du type de notre simulateur.

Fig. 2 : Un simulateur de type « maquette échelle 1 »

Nous avons pu, tout de même, tester sur site les équipements des sociétés KONGSBERG et TRANSAS. Le rendu de la mer reste terne. Les paramètres, tels que le vent, les courants ou la profondeur ont peu d'influence sur le rendu de la mer. Ces dernières années, les avancées ont surtout porté sur la construction de paysages 3D à partir de relevés satellites et sur l'exploitation des données relatives aux courants de marée calculées à partir de modèles numériques.

Seule la société française FAROS [Faros] et sa filiale italienne SINDEL nous ont répondu avec, toutefois, une clause de confidentialité portant sur les réponses. Pour la société FAROS, qui commercialise un simulateur d'entrée de gamme, un rendu physiquement réaliste de la mer n'est pas une priorité. Par contre, c'est une priorité pour sa filiale SINDEL qui commercialise un simulateur haut de gamme de type « maquette échelle 1 ».

1.4 Les autres domaines d'application

Le rendu physiquement réaliste de scènes océanes en temps réel est une préoccupation commune à plusieurs domaines d'application. Comme domaine très proche de la simulation d'entraînement, on peut déjà citer la conception de navire. Mais, à une échelle plus macroscopique, ces deux segments de marché paraissent très étroits dès lors qu'on les compare avec celui des jeux vidéos.

Des publications sur ce thème sont accessibles sur le site internet gamasutra [Gamas] ; ce site est dédié aux développeurs de jeux vidéos. Sur le site internet VTP [VTP], les publications sont triées en deux catégories : les algorithmes de rendu temps réel et ceux qui ne le sont pas. Ce site est très bien organisé. Il présente également des algorithmes de rendu sur d'autres thèmes comme, par exemple, le rendu de terrains qui s'applique, dans notre cas, à la représentation de la côte.

Chapitre 2

Un peu d'océanographie

2.1 Introduction

Nous débutons ce chapitre par une typologie des vagues océaniques. Nous décrivons ensuite le processus de génération et de propagation des vagues océaniques. Nous expliquons alors les phénomènes physiques en présence, en établissant les relations existantes avec les principaux paramètres d'entrée de notre système qui sont le vent, la profondeur et le courant de marée.

2.2 Typologie des vagues océaniques

Définitions et notations :

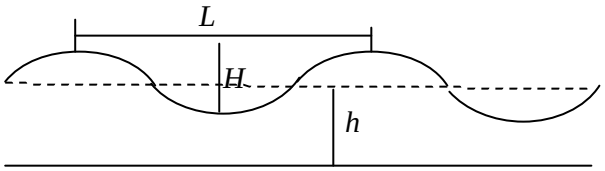


Fig. 3 : Définition d'une vague

. La **surface libre** de l'océan est le niveau théorique de l'océan au repos sans aucune ondulation.

. La **hauteur** *H* de la vague est la différence de niveau entre un creux et une crête successifs. L'**amplitude** a est égale à *H/2*.

. La **période** *T* est le temps qui sépare le passage de deux crêtes successives.

. La **longueur d'onde** *L* est la distance qui sépare le passage de deux crêtes successives.

. La **célérité** *C* de la vague est la vitesse moyenne de propagation des crêtes, égale à *L/T*.

. La **fréquence** *f* de la vague est le nombre de périodes par seconde, égale à *1/T*. Elle est exprimée en hertz.

. On appellera *k* le **nombre d'onde** de la vague égal à *2p/L*.

. On notera *g* la **gravité terrestre**.

. La **profondeur** h est la distance entre la surface libre de l'eau et le sol.

. La **pulsation** w de la vague est le nombre de radians effectués par seconde, égale à $2p/T$. En l'absence de courant, cette grandeur est invariante : nous l'appelons pulsation absolue dans la suite du document et nous la notons w_a. En présence de courant, cette grandeur est encore invariante pour un observateur se déplaçant à la vitesse du courant. Par contre, elle ne l'est plus pour un observateur fixe. La variation qu'il constate est l'effet Doppler. Nous appelons pulsation relative la nouvelle grandeur et nous la notons w_r.

Une première classification des vagues est établie en fonction de leur période, selon Munk dans [Lac65]. On distingue ainsi les **ondes longues** des **ondes de surface**, bien qu'il n'existe pas de différence de nature entre ces deux catégories d'ondes.

Les **ondes longues** commencent vers la période de 5 minutes. Ce sont les ondes dues à la tempête et aux tremblements de terre, et les ondes de marée. Aux périodes de 12 heures environ et de 24 heures environ, on rencontre les périodes des ondes de marée.

On trouve des **ondes de surface** jusqu'à la période de 5 minutes. Parmi elles, les périodes des ondes dues au vent s'étendent jusqu'à environ 30 secondes ; au-delà de 30 secondes, des phénomènes complexes expliquent la présence **d'ondes de surface** de quelques minutes de période. Une **onde de surface** engendre un déplacement temporaire des particules d'eau. Celles-ci suivent une orbite pour revenir à leur position initiale après le passage de la vague.

Concernant les ondes dues au vent, on distingue les **ondes capillaires** des **ondes de gravité**. On les différencie en fonction de leur longueur d'onde (< 1.74 cm). On trouve ainsi des **vagues capillaires** jusqu'à une période de 0.1 s, ayant une longueur d'onde d'environ 1.73 cm et une vitesse de 24 cm/s [Kins65].

2.3 Le processus de génération et de propagation des vagues océaniques

« De tout temps, les marins ont observé la mer pour mieux la comprendre »

Si l'on se réfère au code maritime, toutes les manœuvres sont répertoriées en fonction des conditions météorologiques, depuis le simple fait d'enlever une amarre à quai jusqu'à la manœuvre la plus complexe à exécuter en pleine mer. Les *états de mer* sont classés sur une échelle allant de 0 à 12. Les vitesses du vent et du courant sont exprimées en km/h ou en nœud (*annexe B).*

L'étude successive de la génération et de la propagation des vagues océaniques amène à distinguer ce qui est communément appelé la **mer du vent** de la **houle.**

. Dans le premier cas, la **mer du vent**, un système de vagues est créé sous l'action du vent. Tout d'abord, les vagues ont des périodes, des longueurs d'onde et des hauteurs différentes qui sont des fonctions croissantes de la force, de la durée d'action et de l'étendue sur laquelle souffle le vent générateur. Ensuite dans cette zone dépressionnaire, appelée zone de *fetch*, le vent générateur n'a pas, en un point donné, une direction et une force rigoureusement constante, d'où la naissance de plusieurs trains de vagues qui se propagent dans des directions différentes. La surface de la mer présente alors une allure très hachée, bien que la direction moyenne de propagation de l'ensemble des trains de vagues est globalement celle du vent.

Dans les années 1950, il existait un abîme entre cette agitation incohérente et la **houle** harmonique simple (régulière) étudiée par les mathématiciens. À la fin des années 50, un pont a pu être jeté sur cet abîme, grâce à l'introduction, dans l'étude de l'agitation dans une zone de *fetch*, des méthodes de la statistique. C'est à cette époque que les océanographes Phillips [Phi57] et Pierson et Moskowitz [PM64] ont posé les bases de **l'approche spectrale** .

La première hypothèse est l'hypothèse de linéarisation de tous les mouvements qui autorise à les superposer. D'après le principe de superposition, le profil de la surface de la mer devient une somme de composantes harmoniques (d'ondes de surface) appelée **spectre de vagues**. Chaque onde de surface présente des caractéristiques très différentes.

La deuxième hypothèse peut être exprimée comme suit : à une énergie maximale cédée par le vent correspond une hauteur maximale d'une vague de gravité (de longueur d'onde $\geq$ 1.74 cm). En étudiant les spectres de vagues de gravité générés à l'intérieur d'une zone de *fetch*, dans des conditions de vent données, Phillips [Phi57] a alors proposé une formulation théorique de la hauteur des vagues de gravité en fonction de leur nombre d'onde. Cette formulation est plus connue sous le nom de **spectre d'amplitudes** ou spectre théorique. Sept ans plus tard, Pierson et Moskowitz [PM64] ont également proposé un nouveau spectre d'amplitudes. Aujourd'hui, les systèmes de mesure permettent de vérifier la pertinence de leurs résultats théoriques.

À l'extérieur de la zone de *fetch*, il s'opère un triple triage, qui étale la zone d'agitation à mesure que le temps passe.

Les vagues de courtes longueurs d'onde subissent par viscosité et turbulence des pertes d'énergie plus notables que les lames longues ; elles s'amortissent très rapidement, ce qui confère à la mer une allure beaucoup moins hachée, tandis que les vagues de grandes longueurs d'ondes peuvent traverser les océans pour parvenir jusqu'à nos côtes (fig. 4).

Les tris suivants sont dus à la qualité dispersive du milieu marin au regard des ondes de gravité. À des distances croissantes de la zone de *fetch*, les trains de vagues de gravité qui en proviennent ont des directions de propagation de moins en moins différentes (divergence en direction). D'autre part, plus la longueur d'onde d'une vague de gravité est importante, plus celle-ci se déplace rapidement, occasionnant le phénomène de dissociation (dispersion) des vagues. La célérité d'une vague de gravité dépend de sa longueur d'onde.

Nous sommes alors en présence de la **houle ;** le nombre d'ondes de surface à sommer pour reconstituer la surface de la mer est moins important que celui de la mer du vent. Chaque train de vagues se meut à sa célérité propre. Mais l'ensemble des trains de vagues constitue un groupe de vagues – on parle d'**une houle** – comme il se déplace à sa célérité de groupe propre. Chacune des vagues individuelles paraît progresser dans le groupe, le remonter et disparaître à sa limite antérieure, cependant qu'à l'arrière du groupe une nouvelle vague se forme au même rythme.

Lorsque la vague de plus grande longueur d'onde parvient jusqu'à nos côtes, la houle nous paraît très régulière comme elle est constituée d'un seul train de vagues. Ainsi, plus la zone de *fetch* est éloignée des côtes, plus la **houle** paraît régulière, car moins le nombre de trains de vagues en présence est élevé.

	Force 2 Brise légère	Force 6 Vent fort	Force 10 Tempête
Longueur d'action	0,56 mille	140 milles	1 570 milles
Durée d'action	0,7 heure	15 heures	73 heures
Période des vagues	1,4 seconde	7 secondes	14,7 secondes
Longueur d'onde des vagues	2 mètres	51 mètres	225 mètres
Hauteur des vagues	0,05 mètres	2,5 mètres	15,8 mètres

Fig. 4 : Exemple de caractéristiques des vagues en fonction du vent générateur

En pleine mer, il n'est pas rare que des groupes de vagues créés par des vents générateurs distincts se rencontrent. D'un autre côté, la mer du vent n'est pas un phénomène propre à la haute mer, puisque la zone dépressionnaire peut très bien être située près des côtes.

De manière générale, une vague de gravité se forme pendant la période où le vent lui cède une partie de son énergie. Puis, elle s'amortit durant sa propagation ou aux abords des côtes. Dans ce dernier cas, la vague s'amortit à cause de la dissipation d'énergie due aux frottements des particules d'eau sur le fond de la mer et de la dissipation d'énergie occasionnée lors du déferlement sur le rivage. En l'absence de dissipation, l'énergie de la vague reste constante : à une énergie donnée correspond une hauteur de la vague. Le principe de conservation de l'énergie se vérifie.

Près du rivage, lorsque la profondeur est inférieure à une demie longueur d'onde, les variations de profondeur changent la forme et la direction de propagation des vagues de gravité. Lorsque la profondeur diminue, la longueur d'onde de la vague diminue tandis que sa hauteur augmente ; réciproquement, lorsque la profondeur augmente, la longueur d'onde de la vague augmente tandis que sa hauteur diminue. La célérité augmente ou ralenti en conséquence. Toute variation de la longueur d'onde d'une vague entraîne une variation de sa direction de propagation. C'est le phénomène de réfraction.

2.4 Les courants marins

Si une vague occasionne un déplacement temporaire des particules d'eau qui reviennent à leur position initiale après le passage de la vague, le courant, lui, engendre un transfert horizontal des particules d'eau. En présence de courant, un observateur fixe, qui ne se déplace pas à la vitesse du courant et qui examine un train régulier de vagues se propageant dans sa direction, peut mesurer que le temps écoulé entre les passages successifs de deux crêtes n'est pas le même. Cette variation est l'effet Doppler.

Les effets du courant sur les vagues sont plus subtils que ceux occasionnées par les variations de profondeur à l'abord des côtes [Wol99]. Au mouvement caractéristique d'une onde est additionné un déplacement horizontal de la masse d'eau. Plus la force d'un courant, dont la direction est opposée à la direction de propagation d'une vague, augmente, plus la longueur d'onde de la vague diminue tandis que sa hauteur augmente. La réciproque est vraie. D'autre part, plus la force d'un courant, de même direction que la direction d'une vague, augmente, plus la longueur d'onde de la vague augmente tandis que sa hauteur diminue. En termes marins, on parle de courant arrière. La réciproque est vraie. Toute variation de la longueur d'onde d'une vague entraîne une variation de sa direction. C'est le phénomène de réfraction.

Sans dissipation d'énergie, le principe de conservation de l'énergie peut être appliqué. Par contre, comme l'effet Doppler a pour effet de modifier la période de la vague, nous verrons dans le chapitre 4 que ce principe ne peut pas être appliqué directement. Le cas extrême est le cas d'un déferlement de la vague occasionné par un courant contraire, pour laquelle une partie de son énergie est dissipée (fig. 5).

Fig. 5 : Houle opposée au courant des Aiguilles au large de la côte Sud-Africaine [Ifrem]

Il existe deux catégories principales de courants marins qui sont les **courants océaniques de surface** et les **courants de marée** :

. Les **courants océaniques de surface** sont produits par la différence du bilan radiatif solaire. Le réchauffement ou le refroidissement de la masse d'eau induit change la température et la salinité de

l'eau, ce qui provoque des changements de densité, d'où des déplacements verticaux de la masse d'eau. De plus, le réchauffement ou le refroidissement de la masse d'air génère des vents qui entraînent la surface de l'eau par friction, ce qui donne naissance aux courants océaniques de surface. À cause de la force géostrophique, encore appelée la force de Coriolis (*annexe B)*, les vents créent de larges systèmes circulaires qui tournent dans le sens des aiguilles d'une montre dans l'Atlantique Nord et le Pacifique Nord, et en sens inverse dans l'Atlantique Sud, le Pacifique Sud et l'Océan Indien (fig. 6). La masse d'une particule d'eau est suffisamment faible pour que la force de Coriolis puisse être considérée comme négligeable. Cette force n'a un effet que sur des masses gigantesques correspondant à des tailles caractéristiques de l'ordre de la centaine de kilomètres.

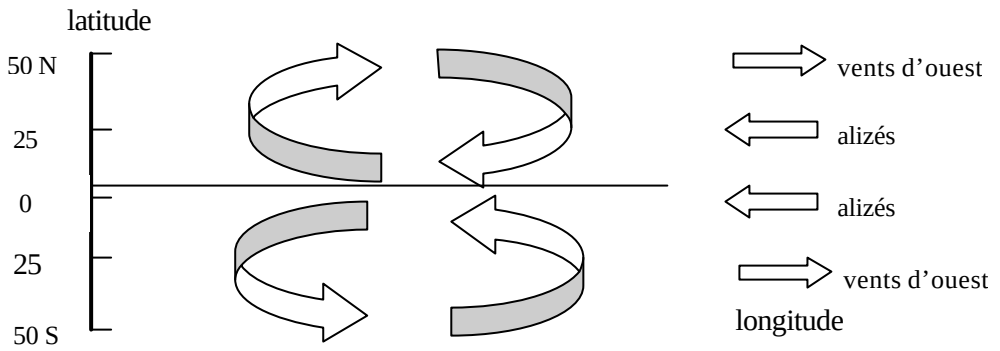


Fig. 6 : Les courants océaniques de surface

. Les **courants de marées** sont des ondes longues. Ils sont dus à la marée d'origine astronomique qui est le facteur le plus important de la variabilité de la surface libre des océans. La marée d'origine astronomique est définie comme étant la variation à allure périodique du niveau des océans due aux seules actions des corps célestes qui se déplacent autour de la terre. C'est le mouvement relatif apparent de la lune et du soleil par rapport à la terre qui, combiné à la rotation de la terre sur elle-même, génère des forces d'attraction gravitationnelle périodiques entraînant un déplacement périodique et ondulatoire des masses d'eau océaniques. La rigoureuse périodicité du phénomène permet l'analyse harmonique de la marée.

En pleine mer, les effets des courants de marées sont presque négligeables devant ceux des courants océaniques de surface alors que, près des côtes, c'est le contraire.

À ce jour, les données se rapportant aux courants de marées sont beaucoup plus fiables que celles que se rapportant aux courants océaniques de surface. Les modèles numériques de courants de marées sont basés sur l'étude théorique de chacune des principales composantes harmoniques. De plus, depuis le lancement du satellite altimétrique TOPEX/Poséidon en 1992, une meilleure précision des données a pu être obtenue grâce aux méthodes d'assimilation des données d'observations [Lef99]. Sur une station de cartographie embarquée à bord d'un navire, il devient ainsi possible de connaître la hauteur d'eau en un point donné, ainsi que la force et la direction du courant de marées. Sur la carte, la force et la direction du courant de marées sont représentées par des vecteurs dont la dimension est proportionnelle à la force du courant (fig. 7).

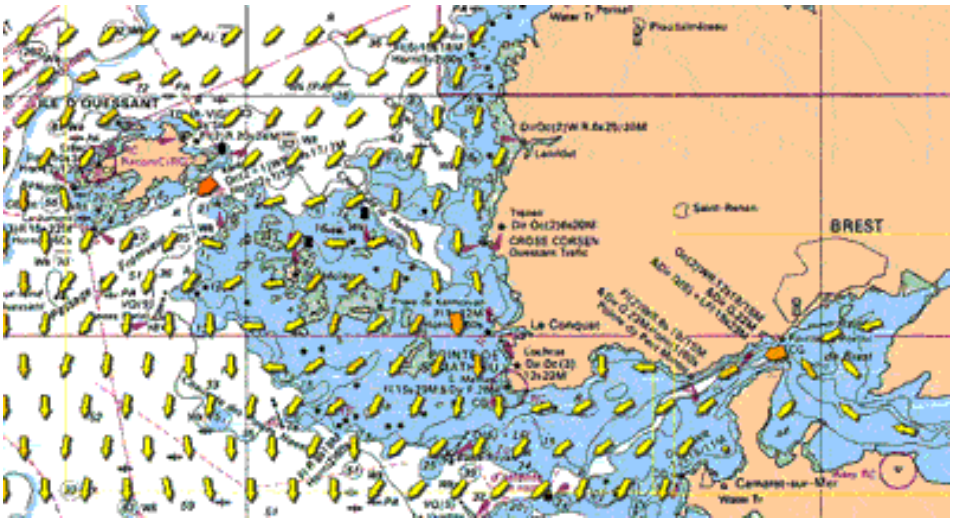


Fig. 7 : Carte des courants de marées [IM]

Chapitre 3

Théories physiques et modélisation des vagues océaniques

3.1 Introduction

On peut classer les théories physiques des vagues océaniques en trois grandes catégories.

Le premier groupe de théories est basé sur une mise en équation fine de la mécanique des fluides [Com90]. Celles-ci furent introduites par Navier-Stokes. Nous parlerons de simulation complète du fluide lorsque nous ferons référence à ce premier groupe de théories. Du point de vue pratique, la résolution de ces équations par la méthode classique des éléments finis pose des problèmes de stabilité numérique et de temps de calculs contraignants. Des hypothèses simplificatrices ont alors été avancées.

Les théories les plus anciennes définissent la surface des vagues comme un mouvement ondulatoire régulier (la houle). Leurs champs d'application supposent que le vent a cessé de souffler et que les vagues, aux abords des côtes, se propagent sur un fond uniforme et faiblement pentu. L'expression mathématique est simple : **une onde de surface** est exprimée sous la forme d'une équation paramétrique. Les hypothèses sont cependant très restrictives : il est supposé que la pression atmosphérique est constante en tout point, et que l'océan est un fluide parfait, homogène, incompressible et pesant. Nous parlerons de simulation d'une onde de surface lorsque nous ferons référence à ce deuxième groupe de théories.

Les théories suivantes sont basées sur l'hypothèse de linéarisation de tous les mouvements qui autorise à les superposer. D'après le principe de superposition, le profil de la surface de la mer devient une **somme d'ondes de surface.** Leur champ d'application est la génération de vagues océaniques sous l'effet du vent. L'introduction des outils de la statistique permet de rendre le cadre d'hypothèses moins restrictif que précédemment. Nous parlerons de simulation spectrale lorsque nous ferons référence à ce troisième groupe de théories.

Après avoir décrit succinctement les principaux résultats théoriques, nous présentons dans ce chapitre les modèles infographiques que nous avons classé par groupe de théories sur lesquels ils s'appuient, et par ordre chronologique afin d'illustrer les apports successifs.

3.2 Théories physiques

3.2.1 Les équations de la mécanique des fluides

Les deux plus importantes lois de la mécanique des fluides à résoudre sont les équations de Navier-Stokes et l'équation de continuité du fluide. À des fins de simplification, nous considérons le

fluide (l'eau de mer) comme étant un fluide visqueux, incompressible et conservatif. Cette dernière considération induit qu'il n'y a pas d'apport ou de retrait de fluide.

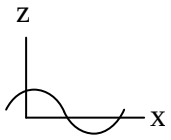
La dynamique de la physique classique est gouvernée par la loi de Newton ($SF = m\ dV/dt$) qui, pour une particule de fluide, prend la forme des équations de Navier-Stokes :

$$\frac{dV}{dt} = F - \frac{1}{?}\nabla P + ??V \qquad (3.1)$$

où ∇ et Δ représentent respectivement les opérateurs gradient et Laplacien et où la viscosité u est une propriété intrinsèque du fluide correspondant à sa résistance au mouvement (résistance au cisaillement).

Cette équation met en relation la dérivée par rapport au temps de la vitesse V avec les forces externes F (vent, pesanteur, ...), les forces internes résultant des variations de pression P pour une densité r, et l'écoulement du fluide fonction de sa viscosité u.

En vue de profil, les deux composantes u,w du vecteur V sont dépendantes du système de coordonnées x,z tandis que le gradient de vitesse se développe classiquement par :



$$\frac{dV}{dt} = \frac{\partial V}{\partial t} + \frac{\partial V}{\partial x}\frac{\partial x}{\partial t} + \frac{\partial V}{\partial z}\frac{\partial z}{\partial t} = \frac{\partial V}{\partial t} + u\frac{\partial V}{\partial x} + w\frac{\partial V}{\partial z} = \frac{\partial V}{\partial t} + V\nabla V$$

où $\P V/\P t$ indique la variation en un point fixe de l'espace.

De plus, l'équation de continuité du fluide traduit le principe de conservation de la masse. La variation de masse du fluide contenu dans un volume donné, durant un laps de temps donné, égale à la somme des masses de fluide entrantes, diminuées de celles sortantes, doit être nulle.

L'eau de mer est incompressible : la densité est constante au cours du temps. L'équation de continuité se simplifie en :

$$div\,V = \frac{\partial u}{\partial x} + \frac{\partial w}{\partial z} = 0 \qquad (3.2)$$

3.2.2 La forme linéaire des équations de Navier-Stokes

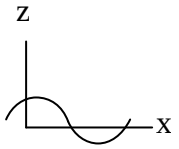
Nous effectuons, ici, une description très sommaire de deux simplifications des équations de Navier-Stokes. Pour plus de détails, nous invitons le lecteur à consulter [Lac65] et [Kin65].

3.2.2.1 Réécriture des équations de Navier-Stokes

Dans cette section, on fait l'hypothèse que le fluide est incompressible et non visqueux. Pour un fluide irrotationnel, les équations de Navier-Stokes s'écrivent en terme de potentiel de vitesse f. Les vitesses des particules dérivent d'un potentiel f :

$$V = \nabla f \qquad (3.3)$$

dont la décomposition, en vue de profil, suivant les axes *x,z* est :



$$\begin{cases} u = -\dfrac{\partial f}{\partial x} \\ w = -\dfrac{\partial f}{\partial z} \end{cases}$$

Pour un fluide incompressible, l'équation de continuité du fluide (3.2) devient :

$$div V = div \nabla f = D f = 0 \qquad (3.4)$$

Si on considère en outre le fluide parfait, l'équation du mouvement (3.1) devient :

$$\frac{\partial \nabla f}{\partial t} + \nabla\left(\frac{1}{2}(\nabla f)^2 + \frac{P}{r} + gz \right) = 0 \qquad (3.5)$$

qui s'intègre pour donner l'équation de Bernouilli, où la constante dépendante du lieu de passage est intégrée dans f :

$$\frac{\partial f}{\partial t} + \frac{1}{2}(\nabla f)^2 + \frac{P}{r} + gz = 0 \qquad (3.6)$$

où les deux premiers termes correspondent à l'énergie cinétique tandis que les deux derniers termes correspondent à l'énergie potentielle.

Condition de pression

Nous appelons h l'élévation de la surface au-dessus du niveau de repos. À la surface, la pression, supposée constante, est égale à la pression atmosphérique. On l'intègre également dans f, d'où :

$$\frac{\partial f}{\partial t} + \frac{1}{2}(\nabla f)^2 + gh = 0 \qquad \text{en } z = h\text{, à la surface} \qquad (3.7)$$

3.2.2.2 Vagues d'amplitude finie

Pour résoudre l'équation (3.7), Sir Georges Stokes a proposé d'utiliser un repère de coordonnées se déplaçant à la vitesse C de la vague. Dans le repère mobile, tout semble immobile (stationnaire) et le terme ¶*f*/¶*t* est nul ; le système d'équations que nous devons résoudre s'écrit :

$$\begin{cases} D f = 0 & \\ \frac{1}{2}(\nabla f)^2 + gh = 0 & \text{en } z = h\text{, à la surface} \\ \frac{\partial f}{\partial z} = 0 & \text{en } z = -h\text{, au fond} \end{cases} \qquad (3.8)$$

Pace qu'il n'est pas possible de trouver de solution analytique à ces équations, une solution est donc de faire un développement limité d'ordre 1 ou à des ordres supérieurs. En poussant les développements limités à des ordres supérieurs, on obtient les différentes vagues non linéaires de Stokes.

Les formules suivantes sont extraites de Aristaghes [AA85] :

Développement limité à des ordres supérieurs

$m = kx - w_r t$

Stokes d'ordre 1.

$$\begin{cases} x_1 = \frac{H}{2}\cos(m) & (3.9) \\ f_1 = \frac{gHT}{4p}\left(\frac{\cosh(k(z+h))}{\cosh(kh)}\right)\sin(m) & (3.10) \\ L = \frac{gT^2}{2p}\tanh\left(\frac{2ph}{L}\right) & (3.11) \end{cases}$$

Stokes d'ordre 2.

$$\begin{cases} x_2 = x_1 + \frac{pH}{4L}\left(\frac{3-\tanh^2(kh)}{\tanh^3(kh)}\right)\cos(2m) \\ f_2 = f_1 + \frac{3p\,H^2}{16T}\left(\frac{\cosh(2k(z+h))}{\sinh^4(kh)}\right)\sin(2m) \\ L = \frac{gT^2}{2p}\tanh\left(\frac{2ph}{L}\right) \end{cases}$$

3.2.2.3 Vagues de faible amplitude

La deuxième famille d'équations est obtenue sous l'hypothèse que les mouvements de l'eau sont faibles, d'où le nom de vagues de faible amplitude. Cela présuppose que la hauteur de la vague est faible devant sa longueur d'onde : H/L << 1 (Hypothèse 1). Cette première hypothèse nous permet de négliger le terme non linéaire de l'équation (3.7). On obtient ainsi la forme linéaire des équations de Navier-Stokes :

$$\begin{cases} Df = 0 & \\ \frac{\partial f}{\partial t} = -gh & \text{en } z = h\text{, à la surface} \quad (3.15) \\ \frac{\partial f}{\partial z} = 0 & \text{en } z = -h\text{, au fond} \end{cases}$$

Condition cinématique

Comme nous avons supposé que la hauteur de la vague est faible devant sa longueur, nous supposons également que la hauteur de la vague est faible devant la profondeur h : H/h << 1 (Hypothèse 2). Si l'on considère que la vitesse de déplacement de la surface libre est sensiblement verticale, on obtient l'équation (3.16) plus connue sous le nom de «condition de Poisson » [Lac65] [Kin65] :

$$\frac{\partial^2 f}{\partial t^2} + g\frac{\partial f}{\partial z} = 0 \qquad (3.16)$$

La condition de Poisson permet d'établir la relation de dispersion d'une vague qui rend compte de l'évolution du nombre d'onde (de la longueur d'onde) en fonction de la profondeur [Lac65] [Kin65] :

$$w_r^2 = gk\ tanh(kh) \qquad \text{en bord de côte} \qquad (3.17)$$

$$w_r^2 = gk \qquad \text{en pleine mer} \qquad (3.18)$$

Dans (3.18), k est le nombre d'onde de la vague située en pleine mer sans effet additionnel dû à la profondeur (*tanh(kh) ® 1*) ; il est noté $k_¥$ dans la suite du document. Près des côtes, à cause des variations de profondeur, le nombre d'onde k devient *ktanh(kh)*. Parce que la pulsation, elle, reste constante, on peut écrire que $k_¥ = ktanh(kh)$. Sans présence de courant, pour vérifier que la pulsation reste constante quelle que soit la profondeur, on peut effectuer l'observation suivante. Soit un train régulier de vagues se propageant vers les côtes, lorsque la longueur d'onde et la hauteur des vagues se voient modifiées suite à un dénivellement local du fond de la mer, tour à tour, leur forme géométrique est modifiée mais il est conservé un intervalle de temps régulier entre deux crêtes successives : les vagues ne se rattrapent pas et reprennent leur forme initiale après le dénivellement. Les océanographes en ont déduit que, en l'absence de courant, la période des vagues restait constante.

Dans ces conditions, la vitesse des vagues dépend de leur longueur d'onde : $C = \frac{w_r}{k} = \sqrt{\frac{g}{k}}$

L'équation (3.18) est identique à l'équation (3.11) relative à la théorie de Stokes au premier ordre. Les solutions de (3.15) sont [Lac65] [Kin65] :

$$h = D\frac{w_r}{g}cosh(kh)cos(kx - w_r t) = acos(kx - w_r t) \qquad (3.19)$$

avec $a = D\frac{w_r}{g}cosh(kh)$

$$f = \frac{ga}{w_r}\frac{cosh(k(z+h))}{cosh(kh)}sin(kx - w_r t) = \frac{w_r a}{k}\frac{cosh(k(z+h))}{sinh(kh)}sin(kx - w_r t) \qquad (3.20)$$

Les équations (3.19) et (3.20) sont similaires à celles de la théorie de Stokes au premier ordre (3.9) et (3.10).

Le champ de vitesse

La valeur du potentiel nous permet de calculer le champ de vitesse des particules, qui est :

$$\begin{cases} u = -\dfrac{\partial f}{\partial x} = a w_r \dfrac{\cosh(k(z+h))}{\sinh(kh)} \cos(kx - w_r t) \\ w = -\dfrac{\partial f}{\partial z} = a w_r \dfrac{\sinh(k(z+h))}{\sinh(kh)} \sin(kx - w_r t) \end{cases} \qquad (3.21)$$

Les vitesses des particules sont tangentes à la crête et au creux. Elles sont dirigées dans le sens de la propagation sur la crête et en sens inverse dans le creux. Un objet à la surface aura ainsi tendance à aller dans le sens de la propagation de la vague lorsqu'il se trouve sur la crête et à revenir en arrière lorsqu'il se trouve dans le creux (fig. 8). Ce sont quelques-uns des effets que nous aurons à prendre en compte lors de l'étude du comportement du navire (chapitre 7). En termes marins, on parle de tenue à la mer du navire.

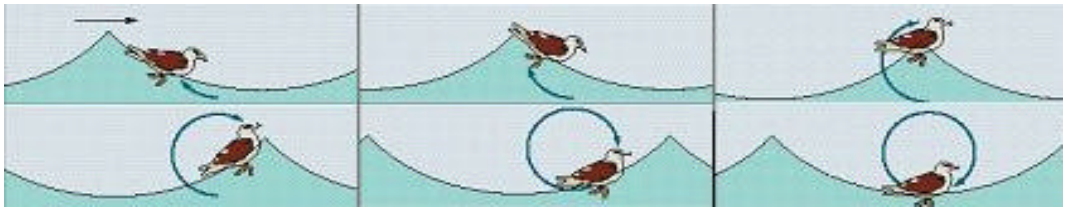
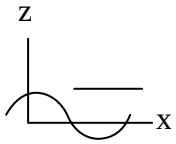


Fig. 8 : Mouvements d'un objet à la surface de l'océan [Ifrem]

Trajectoire des particules

Les trajectoires des particules sont des ellipses fermées parcourues en une période (fig. 9). Comme $u=dx/dt$ et $w=dz/dt$, nous pouvons déterminer les déplacements des particules. Soit, x et y, le déplacement de la particule qui, au repos, se trouve en x_0, z_0 :



$$\begin{cases} x = x_0 - a \dfrac{\cosh(k(z_0+h))}{\sinh(kh)} \sin(k x_0 - w_r t) \\ z = z_0 + a \dfrac{\sinh(k(z_0+h))}{\sinh(kh)} \cos(k x_0 - w_r t) \end{cases} \qquad (3.22)$$

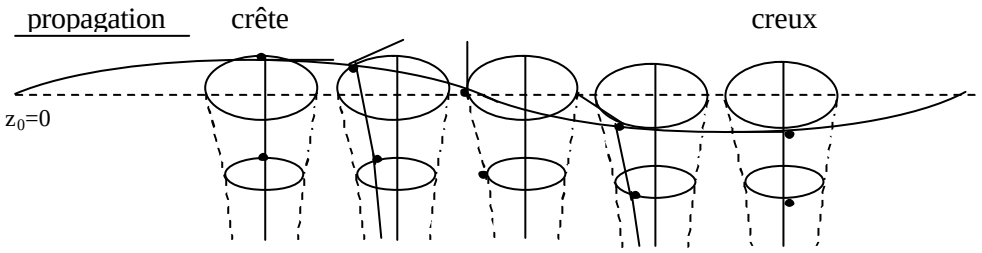


Fig. 9 : Vagues de faible amplitude

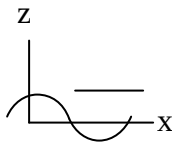
À la surface, la position moyenne des particules est $z_0 = 0$. Il n'y a pas de surélévation de la position moyenne des particules par rapport à la surface libre de l'océan.

Le grand axe et le petit axe de l'ellipse sont égaux à :

$$a\frac{cosh(k(z_0+h))}{sinh(kh)}$$ qui est la longueur du grand axe (axe horizontal).

$$a\frac{sinh(k(z_0+h))}{sinh(kh)}$$ qui est la longueur du petit axe (axe vertical).

Lorsque la profondeur augmente, $tanh(kh)®1$, $e^{-kh}®0$ et $e^{-k(z+h)}®0$, les trajectoires des particules deviennent des cercles parcourus en une période :



$$\begin{cases} x = x_0 - a e^{k z_0} sin(k x_0 - w_r t) \\ z = a e^{k z_0} cos(k x_0 - w_r t) \end{cases} \quad (3.23)$$

La vague de faible amplitude s'appelle aussi la vague d'Airy dans la mesure où George B. Airy a été le premier à obtenir ce résultat.

A partir de la formule (3.17), on peut rechercher quelle est la profondeur qui constitue la limite pratique d'applicabilité des équations (3.23). Si on se fixe d'avoir une précision supérieure à 99 %, on peut adopter comme seuil $L/2$. Quand $h < L/2$, près du rivage, lorsque la vague s'applatit, $sinha®a$ et $cosha®1$, le grand axe de l'ellipse tend vers a/kh.

3.2.3 La théorie de Gerstner

En 1802, le physicien Gerstner a publié une théorie de la vague dans un journal des Carpates qui est restée longtemps ignorée de tous. Pourtant, dans des conditions de profondeur infinie, le résultat obtenu est similaire à celui obtenu par le biais de la forme linéaire des équations de Navier-Stokes. De ce fait, les vagues de Gerstner sont également appelées vagues de Airy.

Lorsque les vagues ont une amplitude finie et que la profondeur devient infinie, la théorie de Gerstner permet de capturer une partie du phénomène physique : chaque particule tourne autour d'un point de coordonnées x_0,z_0 en décrivant une circonférence dont le rayon décroît exponentiellement en fonction de sa distance à la surface (fig. 10). Les circonférences sont inscrites dans un disque dont le rayon est borné par $1/k$.

Trajectoire des particules

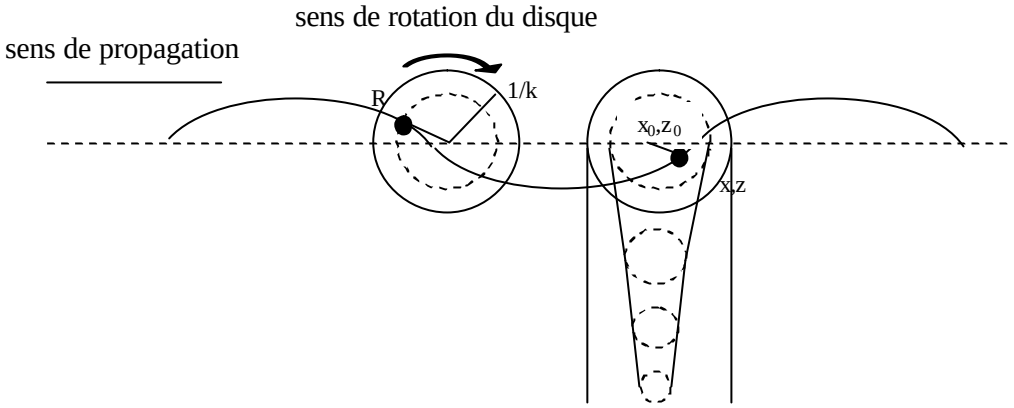
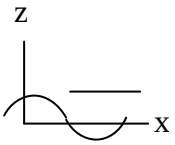


Fig. 10 : Les vagues de Gerstner



$$\begin{cases} x = x_0 + a e^{k z_0} \sin(k x_0 - w_r t) \\ z = z_0 - a e^{k z_0} \cos(k x_0 - w_r t) \end{cases} \qquad (3.25)$$

Les équations (3.25) sont identiques aux équations (3.23).

Pour simplifier, on notera $R = ae^{kz0}$. On peut déduire que, à la surface :

$$R = a e^{kz0} = \frac{H}{2} \qquad (3.26)$$

Les particules en déplacement sont situées sur une *trochoïde* dont l'équation paramétrique est donnée par les équations (3.25). À l'instant t_1, tous les points de même position au repos et de même rayon auront tourné sur leur trajectoire circulaire d'un angle égal à $w_r(t_1-t_0)=2p(t_1-t_0)/T$. La surface sera déplacée dans le sens de rotation des cercles ; la trochoïde représentant la surface libre se déplace donc d'un mouvement de translation non déformé (fig. 11).

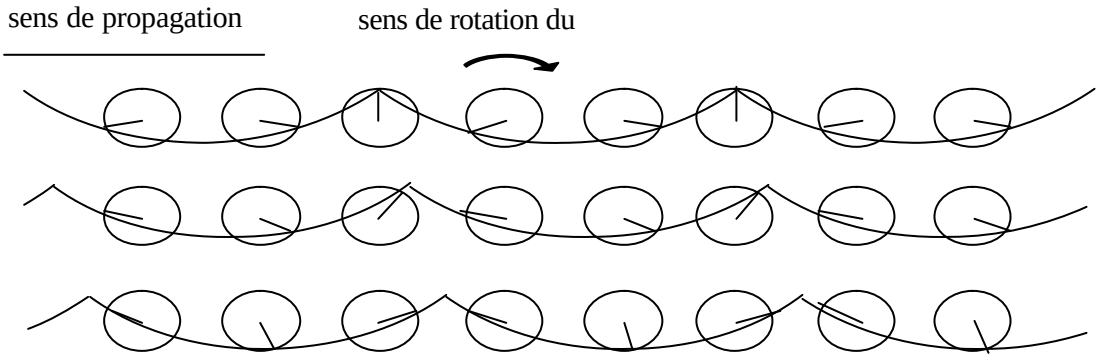


Fig. 11 : Propagation des vagues de Gerstner

Pour produire les six courbe de la figure 12, il a été utilisé comme valeurs respectives de *kR* *0.2, 0.4, 0.6, 0.8, 1.0, 1.2*. On peut constater que :

- Tant que $kR \pounds 1$, la trajectoire des particules correspond à un phénomène naturel.
- Dès que $kR > 1$, la trochoïde comporte une boucle au sommet, ce que les particules ne peuvent faire ; à ce stade, normalement, la vague déferle.

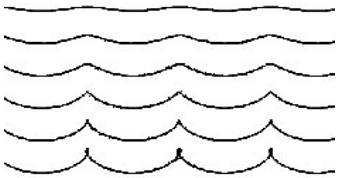


Fig. 12 : Vue de la surface de la mer en fonction de k et de R

Dans la théorie de Gerstner, si on recalcule le niveau moyen des particules, on constate qu'il ne coïncide pas tout à fait avec $z_0 = 0$ contrairement à ce qu'il advient pour la forme linéaire des équations de Navier-Stokes. Il existe une surélévation du niveau moyen par rapport à la surface libre de l'océan.

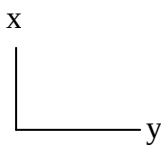
Si l'on applique les équations (3.1) et (3.2) de la mécanique des fluides avec la théorie de Gerstner, on trouve que la surface d'égale pression, quel que soit le temps, est :

$$P = rg\left(\frac{pR^2}{L} - z_0\right)$$

soit le niveau moyen de repos des particules : $$z_0 = \frac{pR^2}{L} = \frac{pH^2}{4L} \qquad (3.27)$$

Le passage en 3D

Dans les livres d'océanographie [Lac65] [Kin65], il nous est proposé une vue de profil d'une onde de surface et les équations sont exprimées en deux dimensions. Pour la transformation de deux à trois dimensions, quelle que soit la profondeur, k devient le vecteur direction de propagation dans le plan de la mer. En rapportant le profil de vague du plan 2D (k / || k ||, z) au repère x,y,z, on obtient :



$$\begin{cases} x = x_0 - a\dfrac{k_x}{\|k\|}\left(\dfrac{\cosh(k(z_0+h))}{\sinh kh(kh)}\sin((k_x x_0 + k_y y_0) - w_r t)\right) \\ y = y_0 - a\dfrac{k_y}{\|k\|}\left(\dfrac{\cosh(k(z_0+h))}{\sinh kh(kh)}\sin((k_x x_0 + k_y y_0) - w_r t)\right) \\ k_x = \|k\|\sin(q) \\ k_y = \|k\|\cos(q) \end{cases} \qquad (3.24)$$

où $(k_x x_0 + k_y y_0)$ est le produit scalaire entre le vecteur nombre d'onde et la position initiale (x_0,y_0), où $\| k \| = 2p/L$ et où les coordonnées du vecteur direction de propagation sont $(\sin(q),\cos(q))$.

La direction de propagation du train de vagues n'est pas constante tout au long du trajet. Elle varie lorsque la longueur d'onde et/ou la période varient ; il s'agit du phénomène de réfraction.

3.3 Modélisation des vagues en infographie

3.3.1 Simulation complète du fluide

3.3.1.1 Le modèle de Foster

La méthode M.A.C. (« Mark-And-Cell method ») [Fle91] est couramment employée en ingénierie pour résoudre les deux équations de la mécanique des fluides (3.1) et (3.2). Elle concerne les fluides ayant une surface libre (surface en contact avec l'air). C'est une méthode de différences finies où l'espace est découpé à l'aide d'une grille 3D régulière. Les deux équations sont appliquées et contrôlées à la frontière de chaque élément de la grille. La résolution se fait en plusieurs passes afin d'obtenir une situation d'équilibre de part et d'autre de ces frontières. En fait, le calcul des dérivées partielles est obtenu par simple soustraction de part et d'autre de la frontière.

La méthode MA.C. a été complètement reproduite en informatique graphique par Foster [FM96]. Elle permet de modéliser les mouvements de tous les types de fluides. Le principe général est le suivant : dans un premier temps, une initialisation des «voxels » (cellules de la grille 3D) est effectuée (vecteurs vitesse, pression au centre). Pour chaque pas de temps, un recalcul des vecteurs vitesse à la frontière de chaque « voxel » est effectué en résolvant les équations de la mécanique des fluides (3.1). L'ensemble de ces nouvelles vitesses ne permet pas de garantir la conservation de la masse et donc de vérifier que l'équation de continuité du fluide est satisfaite (3.2). Pour cela, par un jeu d'aller-retour dans les «voxels », les pressions et les vitesses sont modifiées localement pour converger vers un équilibre de l'équation de conservation de masse. Mais, lorsqu'on modifie localement les vecteurs vitesse d'un « voxel », on modifie les vecteurs vitesse des « voxels » voisins ; il faut alors itérativement modifier l'équilibre pression-vitesse jusqu'à ce que le système se stabilise.

Le vecteur vitesse de chaque particule incluse dans un « voxel » est calculé par interpolation linéaire entre les vecteurs vitesse définis sur les faces du « voxel ».

Le problème inhérent à la méthode réside dans le fait que chaque particule ne doit pas se déplacer de plus d'un « voxel » à chaque pas de calcul afin d'éviter toute divergence des équations. Foster a alors proposé deux méthodes différentes pour contourner ce problème [FM96] et [FM97]. Pour plus de détails, nous invitons le lecteur à se reporter à ces deux articles.

L'autre problème est l'obtention des vecteurs vitesse lorsqu'il existe une interface entre liquide et air dans un même « voxel ». Il peut alors se produire 64 situations de discontinuité selon les faces concernées que Foster a décrit.

Dans ces conditions, à une large échelle, il est difficilement envisageable de procéder à une simulation complète du fluide en temps réel.

3.3.1.2 Le modèle de Chen et lobo

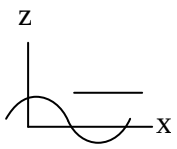
La méthode proposée par Chen et lobo [CL95] consiste à résoudre les équations de la mécanique des fluides (3.1) et (3.2) uniquement à la surface du liquide. La grille permettant de stocker les différents vecteurs vitesse est une grille 2D où seulement deux composantes des vecteurs vitesse sont sauvegardées.

Le calcul des vecteurs vitesse se fait comme pour la méthode M.A.C.. Par contre, pour satisfaire l'équation de conservation de masse, plutôt que de faire varier la pression de la cellule, il est choisi de faire varier la hauteur de la colonne de fluide. Si la divergence du vecteur vitesse de la cellule est positive, donc s'il y a augmentation de masse, la hauteur de la colonne d'eau de cette cellule va augmenter. Ainsi, avec un simple algorithme 2D, on peut représenter une surface en 3D d'un liquide.

Cet algorithme peut être exécuté en temps réel mais l'influence du sol sur la propagation du liquide dans la scène ne peut être traitée que partiellement ; on se contente, ici, de faire varier les hauteurs des colonnes d'eau.

3.3.1.3 Le modèle de layton

A. Layton [Lay02] a proposé très récemment une méthode semi-Lagrangienne (méthode hybride entre une méthode Lagrangienne et une méthode Eulerienne) pour résoudre les équations dites « équations en eaux peu profondes » définies par Haltimer et Williams en 1980 [HW80]. Les auteurs ont simplifié l'équation de la mécanique des fluides (3.1) en limitant leur champ d'application à des fluides sans viscosité. Puis, en 1990, Kass et Miller [KM90] ont formulé l'hypothèse que les vitesses des particules variaient suffisamment lentement pour pouvoir négliger les termes non linéaires. Le système d'équations à résoudre en deux dimensions seulement, en vue de profil, devient :



$$\begin{cases} \dfrac{\partial u}{\partial t} + g\dfrac{\partial h}{\partial x} = 0 \\ \dfrac{\partial w}{\partial t} + g\dfrac{\partial h}{\partial z} = 0 \\ \dfrac{\partial h}{\partial t} + d\left(\dfrac{\partial u}{\partial x} + \dfrac{\partial w}{\partial z}\right) = 0 \end{cases} \tag{3.28}$$

où h est le champ de hauteur et $d = h - b$ avec b qui est la hauteur du fond de la mer (fig. 13).

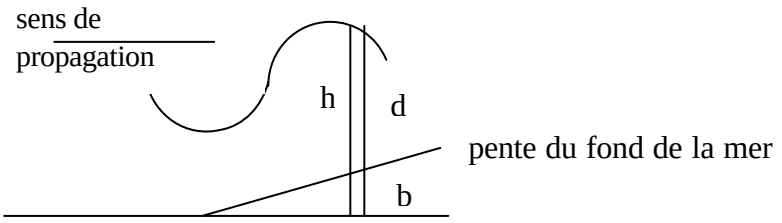


Fig. 13 : Équations en eaux peu profondes

Le modèle d'A. Layton paraît certes séduisant. Les phénomènes de réfraction, réflexion et réfraction sont traités. Par contre, le modèle reste encore difficilement exploitable en temps réel à une large échelle. Si on se réfère aux tests de performance indiqués par A. Layton, la même animation (sans le rendu) d'une surface de fluide de 40 m sur 40 m, 80 sur 80 m et de 100 m sur 100 m a pris respectivement 12.5 s, 55.7 s et 5 min sur un PC doté d'un processeur Pentium III 600 MHz.

3.3.2 Simulation d'une onde de surface

3.3.2.1 Le modèle de Fournier et Reeves

Un des aspects les plus visuels relatifs à la forme de la vague près des côtes a été expliqué par le physicien Biesel [Bie52] : près du rivage, le grand axe de l'ellipse devient parallèle à la pente du fond de la mer (fig. 14).

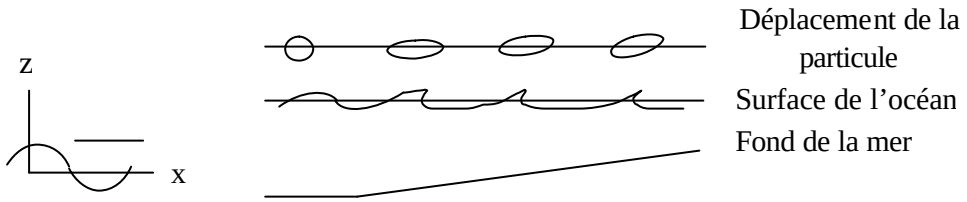


Fig. 14 : Théorie de Biesel

La théorie de Biesel [Bie52] a été adaptée en infographie par Fournier-Reeves [FR86]. Pour une onde de surface de vecteur nombre d'onde k, leur modèle permet de déterminer, en un point initial $X_0 = (x_0,y_0)$ inscrit dans le plan de la mer et de hauteur initiale z_0, sa nouvelle position $X = (x,y)$ dans le plan de la mer et sa hauteur z :

$$\begin{cases} X = X_0 + \frac{k}{\|k\|}\left(\frac{H}{2}\cos(b)S_x\sin(m) + \frac{H}{2}\sin(b)S_z\cos(m)\right) \\ z = z_0 - \frac{H}{2}\cos(b)S_z\cos(m) + \frac{H}{2}\sin(b)S_x\sin(m) \\ S_x = \frac{1}{1-e^{-K_x h_0}},\ S_z = S_x\left(1-e^{-K_z h_0}\right) \\ \sin(b) = \sin(g)e^{-K_0 h_0} \\ m = -w_r t + \sum_0^{x_0} k(h)Dx \\ k(h) = \frac{k_\infty}{\sqrt{\tanh(k_\infty h)}} \end{cases} \qquad (3.29)$$

où S_x et S_z sont respectivement le grand axe et le petit axe de l'ellipse, b et γ sont respectivement les pentes de l'ellipse et du fond de la mer, K_0, K_x et K_z sont trois coefficients arbitraires qui peuvent être fixés, h_0 est la profondeur au point (x_0,y_0) tandis que h est la profondeur en un point (x,y).

Dans certains articles ([T&B87], [Tes01-1], [Gol01], ...) le vecteur nombre d'onde est présent dans les équations alors que dans d'autres articles il ne l'est pas ([F&R86], [Gon99], ...). Jusqu'à la fin du document, nous avons choisi de le faire apparaître pour faciliter la compréhension de l'approche de simulation spectrale (paragraphe 3.3.3).

Pour tenir compte des variations continues de la profondeur le long de la surface à modéliser, les auteurs ont modifié le calcul de la phase en tenant compte des variations continues de k depuis le large :

$$m = -w_r t + \sum_{0}^{x} k(h) Dx$$

Les auteurs ont aussi proposé une approximation à 5% du nombre d'onde près des côtes :

$$k(h) = \frac{k_\infty}{\sqrt{tanh(k_\infty h)}} \qquad (3.30)$$

Leur modèle s'appuie sur la théorie de Gerstner. Concernant la trajectoire des particules d'eau, il est appliqué une matrice de rotation, exprimée à l'aide des angles d'Euler, aux équations (3.25). Le rendu obtenu est très réaliste comme, au fur et à mesure que la vague se rapproche du rivage, le grand axe de l'ellipse s'oriente progressivement par rapport à la pente du sol et que la forme de la vague est alors proche de celle d'un déferlement. Cet aspect visuel continue de susciter beaucoup d'intérêt de la part de la communauté scientifique en synthèse d'images.

3.3.2.2 Le modèle de Gonzato

Le modèle de Fournier et de Reeves s'appuie sur la théorie de Gerstner. Seulement, nous avons vu que la théorie de Gerstner était exacte en pleine mer uniquement, lorsque la profondeur devenait infinie, ce qui pouvait occasionner un rendu irréaliste aux abords du rivage. À ce sujet, dans son mémoire de thèse, J.C. Gonzato [Gon99] a notamment fait remarquer qu'à l'approche du rivage le grand axe de l'ellipse $\frac{H}{2} cos(b) S_x$, tel qu'il avait été défini par Fournier et Reeves, tendait rapidement vers l'infini. Or, dans la théorie de Gerstner, la taille maximale du lieu décrite par les particules doit être inscrite dans un disque dont le rayon est $1/k$. Dans le cas où la trajectoire des particules quitte le disque, il se forme une boucle au sommet de la vague. Pour éviter ce cas, il faut affecter au grand axe de l'ellipse la valeur limite supérieure de $1/k$.

Concernant la théorie des vagues de faible amplitude (paragraphe 3.2.3.3), la condition portant sur le grand axe de l'ellipse est respectée tant que la hauteur de la vague est plus petite que la profondeur.

J.C. Gonzato a également étudié le phénomène de réflexion et celui de diffraction des vagues près des côtes. Quelques phrases ne suffiraient pas ici à décrire convenablement ces deux phénomènes. Aussi, nous invitons le lecteur à se reporter à [Gon99] où une description complète est proposée.

Des valeurs ont été également proposées pour les coefficients arbitraires K_0, K_x et K_z du modèle de Fournier-Reeves. Pour une onde de surface de vecteur nombre d'onde k, son modèle permet de déterminer, en un point initial $X_0 = (x_0,y_0)$ inscrit dans le plan de la mer et de hauteur initiale z_0, sa nouvelle position $X = (x,y)$ dans le plan de la mer et sa hauteur z :

$$\begin{cases} X = X_0 + \frac{k}{\|k\|}\left(\frac{H}{2}t'_b S_x \sin(m) + \frac{H}{2}t_b S_z \cos(m)\right) \\ z = z_0 - \frac{H}{2}t'_b S_z \cos(m) + \frac{H}{2}t_b S_x \sin(m) \\ t_b = |\sin(g)|e^{-0.1h_0}, t'_b = \sqrt{1-t_b^2} \\ S_x = \frac{1}{1-e^{-0.11h_0}}, S_z = S_x\left(1-e^{-0.09h_0}\right) \\ m = -w_r t + \sum_0^{x_0} k(h)Dx \\ k(h) = \frac{k_\infty}{\sqrt{\tanh(k_\infty h)}} \\ S_x \le \frac{1}{\frac{H}{2}k\cos(b)} \end{cases} \quad (3.31)$$

où S_x et S_z sont respectivement le grand et le petit axe de l'ellipse, b et g sont respectivement les pentes de l'ellipse et du fond de la mer. **Le modèle dépend logiquement de la profondeur**. h_0 est la profondeur au point (x_0,y_0) tandis que h est la profondeur en un point (x,y).

Au cours de la simulation, la valeur de S_x est toujours calculée en premier au cas où il faudrait luit affecter la valeur limite supérieure de $\frac{1}{\frac{H}{2}k\cos(b)}$. La valeur de S_z est ensuite déduite de la valeur de S_x.

Nous avons montré dans [Cie01] qu'il était possible d'utiliser le modèle (3.31) en pleine mer et près des côtes. En effet, lorsque la profondeur augmente, on peut remarquer que :

S_x®1, S_z®S_x, t_b®0, t'_b®1 et $\tanh(k_\infty h)$®1. On obtient, alors, le modèle (3.25) pour lequel les particules d'eau décrivent une circonférence.

Dans ce modèle, les effets du courant sur les vagues ne sont pas pris en considération.

3.3.2.3 L'algorithme de lancer de vague

Jusqu'à maintenant, nous savons modéliser la forme d'une vague dont la direction de propagation est constante. Or, chaque fois que la longueur d'onde d'une vague varie, sa direction varie également ; il s'agit du phénomène de réfraction. Les changements successifs de la direction de propagation d'une vague (fig. 15) suivant les axes x et y sont calculés à l'aide de la loi de Snell :

$$\frac{\sin(a_1)}{L_1} = \frac{\sin(a_2)}{L_2} \tag{3.32}$$

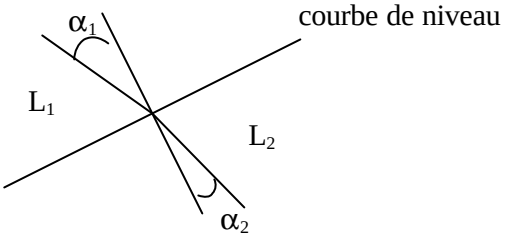


Fig. 15 : Le phénomène de réfraction

Le modèle de Ts'o et Barsky

L'algorithme du lancer de vague a été adapté en infographie par Ts'o et Barsky [TS87]. Le principe général est le suivant : les orthogonales au front de la vague initial sont représentés par des rayons dont on calcule la progression dans le plan de la mer en lançant de nouveaux rayons (fig. 16). Lors du franchissement d'une courbe de niveau (fig. 15), la forme géométrique et la direction de propagation de la vague sont recalculées (direction du rayon en cours de traitement).

Les pré-calculs sont sauvegardés dans une grille 2D qui recouvre la surface à modéliser. Les données sauvegardées sont ensuite exploitées en temps réel. En un point donné, les paramètres pour déterminer la forme et la direction de la vague sont calculés par interpolation à partir des données pré-calculées. D'une image à l'autre, en un même point, seul le temps aura varié. Dans la suite du document, nous appelons *patch Lancer de vague* la surface de la mer associée à une grille donnée. Un *patch* est implémenté par un tableau à deux dimensions.

Le modèle de Gonzato

J.C. Gonzato a proposé une version adaptative de l'algorithme de Ts'o et Barsky dans [Gon99]. Lors de la propagation, il propose d'ajouter de nouveaux rayons lorsque deux rayons consécutifs s'écartent de plus de 1 m, ce qui permet d'augmenter la précision des résultats. Sur la figure 16, cela se traduit par une densité plus importante de rayons entre la figure du milieu et celle de gauche, et une représentation plus réaliste d'un train régulier de vagues pénétrant dans une baie (fig. 16 à droite).

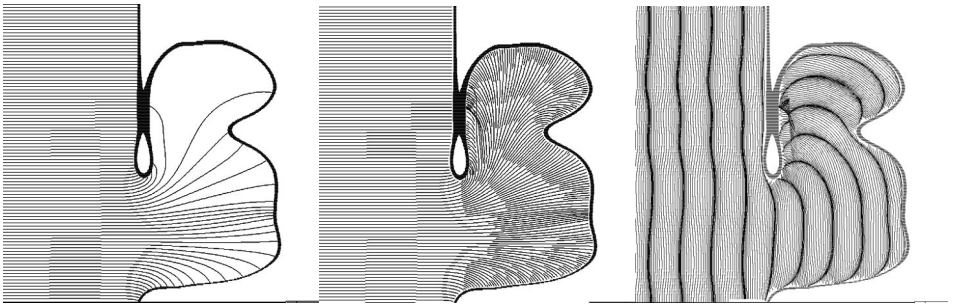


Fig. 16 : L'algorithme du lancer de vague

Dans les articles d'infographie que nous avons pu lire, le front initial d'un train régulier de vagues est représenté par une ligne droite ; cela n'est pas réaliste. Suite à un impact, une vague se propage en décrivant un cercle qui s'éloigne du point d'impact. Une onde de surface due au vent se propage en arc de cercle en s'éloignant de la zone du vent générateur.

D'autre part, il est supposé, ici, que la période du train de vagues ne varie pas, ce qui n'est pas vrai en présence de courant. Lorsque la période varie, il se produit également un changement de la direction de propagation.

3.3.3 Simulation spectrale

3.3.3.1 Le modèle de Mastin

L'approche spectrale, basée sur le principe de linéarisation de tous les mouvements qui autorise à les superposer et sur le principe de définition de la hauteur des vagues en fonction de l'énergie cédée par le vent, a été adaptée en infographie par Mastin [Mas87].

La caractérisation d'un *état de mer* est exprimée, paramétrée et résolue dans l'espace spectral qui est le domaine des fréquences. Un spectre de vagues est une somme de fonctions sinusoïdales périodiques ; chaque fonction est une onde de surface. L'amplitude d'une onde de surface est définie en fonction de sa fréquence (de son nombre d'onde) ; on parle de spectre d'amplitudes.

Mastin se base sur la méthode proposée par Hasselmann [Has80] pour étendre le spectre théorique de Pierson et Moskowitz [PM64] de 1 dimension à 2 dimensions. Une unique FFT inverse produit le champ des hauteurs.

Dans l'article, le paramètre temps n'a pas été introduit dans le modèle, ce qui en limite l'intérêt pour un simulateur.

3.3.3.2 Le modèle de Tessendorf

Depuis que le modèle de Jerry Tessendorf a été utilisé dans les films « Titanic », « Le cinquième élément », « WaterWorld » et bien d'autres films très célèbres, et qu'il l'a présenté en 1999 [Tes01-1], ce modèle a fait le tour de la planète et on trouve de nombreuses implémentations. Le modèle est basé sur la théorie linéaire des vagues de faible amplitude et sur l'utilisation de la FFT. Nous avons ajouté l'*annexe C* car nous avons pu constater que les implémentations existantes comportent de nombreuses erreurs lorsque les vagues sont sommées : les hauteurs des vagues sont définies dans une grille dont les bornes sont comprises entre $-N$ et N alors que la somme est calculée dans une grille dont les bornes sont comprises entre 0 et $2N-1$, les hauteurs des vagues ne sont pas normées, …

Comme première propriété, une somme de fonctions sinusoïdales périodiques sur L sera également une fonction périodique sur L :

$$f(x) = a_0 + \sum_{n=1}^{\infty}\left(a_n cos(2pnx/L) + b_n sin(2pnx/L)\right) \qquad (3.33)$$

que l'on peut réécrire comme une somme de nombres complexes et de nombres complexes conjugués :

$$f(x) = a_0 + \sum_{n=1}^{\infty}\left(\left(\frac{a_n - ib_n}{2}\right)e^{i2pnx/L} + \left(\frac{a_n + ib_n}{2}\right)e^{-i2pnx/L}\right) = \sum_{n=-\infty}^{\infty}\left(c_n\right)exp(i2pnx/L) \qquad (3.34)$$

avec $c_{-n} = c_n^*$ qui est le nombre complexe conjugué du nombre complexe c_n.

Le domaine des fréquences est le domaine des points : $k = 2pn/L$ où $n = ..., -2, -1, 0, 1, 2, ...$ dans lequel la décomposition, autorisée par le principe de linéarisation des mouvements, est effectuée. Cette superposition est la transformée de Fourier discrète inverse :

$$f_{FD}(x) = \sum_k \tilde{f}(k)\, exp(ikx) \qquad (3.35)$$

où $\tilde{f}(k)$ (c_n de (3.34)) est l'amplitude du nombre d'onde k appartenant aux domaines des fréquences et où la partie exponentielle est la phase non temporelle.

Grâce à la stratégie de l'algorithmique « diviser pour régner », la complexité d'une sommation peut être ramenée de $Q(N^2)$ à $Q(N/2log_2N)$ où N est le nombre de vagues à sommer. On parle alors de l'algorithme de la transformée de Fourier rapide (FFT), plus rapide en temps d'exécution.

Dans son modèle, J. Tessendorf a introduit le paramètre temps [Tes01-1], ce que Mastin n'avait pas fait [Mas87]. La FFT inverse permet de calculer, en un point initial $X_0 = (x_0,y_0)$ inscrit dans le plan de la mer et de hauteur $z_0 = 0$ (c'est le choix qui a été fait ici), sa nouvelle position $D = (x,y)$ dans le plan de la mer et sa hauteur h :

$$\begin{cases} D(X_0,t) = X_0 + l \sum_k -i\left(\frac{k}{\|k\|}\right)\tilde{h}(k,t)exp(i\, k.X_0) \\ h(X_0,t) = \sum_k \tilde{h}(k,t)exp(i\, k.X_0) \end{cases} \qquad (3.36)$$

où t est le temps, k est le vecteur nombre d'onde de composantes ($k_x=2pn/L_x$, $ky=2pm/L_y$) avec n et m qui sont deux nombres entiers de bornes $-N$ £ $n,m < +N$, et où l est un paramètre global introduit par J. Tessendorf pour contrôler les déplacements dans le plan de la mer ; les vagues semblent alors moins arrondies.

Aux domaines des fréquences de l'espace spectral, l'ensemble discret des points $(k_x ,k_y) = (2pn/L_x ,2pm/L_y)$ dont la répartition constitue une grille 2D, correspond l'ensemble des points $(x_0,y_0) =$

$(nL_x/N, mL_y/N)$ du plan de la mer dont la répartition constitue également une grille 2D. À chaque pas de simulation, $\tilde{h}(k,t)$ détermine le profil pour chaque nombre d'onde k, en chacun des points de coordonnées (k_x, k_y) de la grille 2D de l'espace spectral.

Une première FFT inverse permet d'obtenir le champs des hauteurs $h(X_0,t)$ en les points (x_0,y_0), mais cela ne suffit pas pour caractériser une sommation d'ondes de surface. Nous avons vu, dans le chapitre précédent, qu'il fallait également tenir compte des déplacements dans le plan de la mer. On renouvelle alors une première fois l'opération suivant l'axe x et une deuxième fois suivant l'axe y. Au total, trois FFT inverses sont utilisées (3.36).

Nous avons vu que l'algorithme FFT est performant. Ses autres spécificités doivent être également soulignées.

Tout d'abord, **toute variation de la direction de propagation d'une vague à l'intérieur d'un *patch FFT* est impossible**. L'approche spectrale est une approche collective et statistique où on ne peut ni représenter, ni simuler le comportement particulier d'une vague individualisée ; sa direction de propagation est, par définition, fixe et homogène dans tout le domaine des fréquences.

Ensuite, la FFT inverse produit par construction un **motif périodique** qui peut être répété continûment. Dans ces conditions, on comprend aisément le succès que connaît ce modèle auprès des développeurs d'applications temps réel. À chaque pas de simulation d'une animation, seulement un à trois calculs matriciels sont exécutés pour reproduire tout un plan d'eau. Par contre, l'approche ne peut être physiquement réaliste que pour les zones du plan d'eau qui correspondent aux caractéristiques des vagues choisies pour la simulation.

Le champ d'amplitudes dépendant du temps dans le domaine des fréquences est déterminé par :

$$\tilde{h}(k,t) = \tilde{h}_0(k)\,exp(i\omega_r(k)t) + \tilde{h}_0^*(-k)\,exp(-i\omega_r(k)t) \qquad (3.37)$$

À chaque pas de simulation, on incrémente t et on calcule $\tilde{h}(k,t)$ pour chaque nombre d'onde k. L'ensemble des valeurs calculées sont les paramètres d'entrée de la FFT.

Il faut noter que l'équation (3.37) conserve la propriété de conjugaison des nombres complexes : on peut vérifier que $\tilde{h}(-k,t) = \tilde{h}^*(k,t)$. En conséquence, concernant les paramètres de sortie de la FFT, ce sont toujours des nombres complexes dont la partie imaginaire est nulle.

Pour déterminer l'amplitude initiale des vagues dans le domaine des fréquences, on utilise souvent le spectre de vagues de Phillips [Phi57] :

$$P_h(k) = A\frac{exp\left(-1/(kL)^2\right)}{k^4} \qquad (3.38)$$

où $L=\|W\|^2/g$ est la plus longue vague lorsque la vitesse du vent est l'intensité du vecteur W exprimée en m/s.

Les autres spectres théoriques de vagues ont été définies par Pierson-Moskowotiz [PM64] et Hasselmann [Has80]. Les océanographes ont longuement étudié le spectre des vagues dans une zone de *fetch* dans une direction donnée, celle du vent. Ils en ont extrait une formulation de l'amplitude des vagues exprimée en une dimension : $k = 2pn/L$ où $n = ..., -2, -1, 0, 1, 2, ...$

Pour définir l'amplitude initiale en chacun des points de la grille 2D de l'espace spectral, les océanographes utilisent le produit scalaire entre le vecteur direction du nombre d'onde

$\left(\frac{k_x}{\|k\|},\frac{k_y}{\|k\|}\right)$ et le vecteur direction du vent $\left(\frac{W_x}{\|W\|},\frac{W_y}{\|W\|}\right)$.

De cette manière, conformément aux observations océanographiques, l'amplitude d'une vague décroît au fur et à mesure que la direction de son nombre d'onde s'éloigne de celle du vent. De plus, pour respecter le phénomène naturel, on assigne la valeur 0 à $P_h(k)$ dès lors que le résultat du produit scalaire est négatif ou nul. On évite ainsi de voir se propager une vague perpendiculairement ou dans le sens opposé à la direction du vent.

J. Tessendorf propose de multiplier l'équation (3.38) par $exp(-kl^2)$ pour supprimer les vagues dont la longueur d'onde est très « petite » $l<<L$ (de longueur d'onde < 1.74 cm si on se réfère à des vagues capillaires mais il n'y a pas de définition de proposée dans l'article), pour lesquelles le modèle s'applique mal et dont le traitement risque d'aliaser avec le pas de la grille. Comme l'équation (3.38) ne s'applique qu'à des vagues de gravité, on élimine ainsi leur domaine de fréquence (l'amplitude initiale tend vers 0). Par contre, J. Tessendorf ne propose pas de spectre de substitution pour ces types de vagues.

La dernière opération consiste à définir l'amplitude des vagues lorsque le paramètre temps est égal à 0, soit pour $\tilde{h}_0(k)$:

$$\tilde{h}_0(k) = \frac{1}{\sqrt{2}}(x_r + ix_i)\sqrt{P_h(k)} \qquad (3.39)$$

où x_r et x_i sont deux nombres aléatoires produits par un générateur de nombres aléatoires. À cet effet, J. Tessendorf propose d'utiliser une loi normale centrée réduite de moyenne 0 et d'écart type 1.

Dans ce modèle, les effets du courant et de la profondeur sur les vagues ne sont pas pris en considération.

Chapitre 4

Extensions et apports sur la modélisation de vagues océaniques

4.1 Introduction

Nos apports sur la modélisation des vagues océaniques sont présentés dans ce chapitre.

Nous commençons par introduire les effets des courants marins sur les vagues. À notre connaissance, ces effets n'ont jamais été introduits dans le cadre de la simulation de mer en images de synthèse. Pourtant, le courant est un paramètre essentiel. Par exemple, dans notre cas, l'homme à la barre doit réguler la vitesse du navire en fonction des conditions de courant. Or, lors d'une session de simulation, nous disposons de peu de moyens visuels pour lui rendre compte de la situation. L'impact du courant sur les vagues (sur la surface de la mer) en est un.

Jusqu'à présent, l'algorithme de lancer de vague a été appliqué près de la côte. Il s'agissait principalement de modéliser la course d'une vague aux abords du rivage et, dans ces conditions, le front initial de vague a été défini comme étant une ligne droite ; il est supposé que l'aire génératrice est suffisamment éloignée pour cela. Pourtant, la direction de propagation d'une vague peut également varier en pleine mer. C'est le cas, par exemple, lorsque les conditions de courant changent. Nous appliquons alors l'algorithme de lancer de vague quel soit le lieu géographique.

Après avoir constaté que la somme classique de vagues est inexploitable en temps réel, nous proposons une extension du modèle de J. Tessendorf afin d'y introduire les effets du vent et du courant de marée sur les vagues.

Enfin, nous proposons un modèle de vagues de courtes longueurs d'onde basé sur la FFT. À ce propos, nous définissons un spectre théorique.

4.2 Simulation d'une onde de surface

4.2.1 Interactions vague-courant

Pour débuter ce paragraphe, nous nous situons dans un contexte de profondeur infinie, plus exactement lorsque la profondeur est supérieure à une demie longueur d'onde de la vague, et nous fixons notre cadre d'hypothèses qui est d'ailleurs général à l'ensemble des études que nous devons mener. Les conditions de vent sont stationnaires tandis que les variations de courant (comme les variations de profondeur) s'opèrent graduellement. Nos calculs ne sont plus valables lors de cas extrêmes. Par exemple, lorsqu'un fleuve se jette dans la mer et que le courant induit est capable de bloquer la propagation des vagues. Seul les courants de marée, pour lesquels nous disposons de données fiables, sont d'ailleurs pris en compte (chapitre 2).

Nous avons vu dans le chapitre 2 que les effets des courants sur les vagues étaient plus subtils que ceux de la profondeur sur les vagues. Tout d'abord, les hauteurs et les longueurs d'onde varient suivant la direction du courant. L'autre différence porte sur la période (la pulsation) de la vague qui restait constante lorsque la profondeur diminuait.

Lorsqu'un train de vagues de même pulsation absolue (hors courant) rencontrent un courant opposé, leur pulsation relative w_r augmente parce que leur période relative diminue. Lorsqu'un train de vagues de même pulsation absolue sont portées par un courant arrière, leur pulsation relative w_r diminue parce que leur période relative augmente. Nous sommes en présence de l'effet Doppler (4.1). L'effet Doppler est généralement expliqué à l'aide d'illustrations portant sur les ondes sonores, ce qui en facilite la compréhension. Toutefois, son application est générale à toutes les ondes :

$$w_a = w_r + k.U \qquad (4.1)$$

où U est le vecteur courant, avec, comme direction celle du courant, et comme intensité sa vitesse exprimée en mètres par seconde et k le vecteur nombre d'onde.

Si nous appelons $k_{¥}$ le nombre d'onde du train de vagues en pleine mer sans courant, et T_a la période absolue du train de vagues, la variation de la pulsation intrinsèque en un point x est :

$$w_r(U) = \frac{2p}{T_a} - \|k_\infty\| \|U\| cos(a) \qquad (4.2)$$

où a est l'angle que forment les vecteurs nombre d'onde et courant, et où $\| U \|$ est la vitesse du courant au point x qui est exprimée en m/s.

Parce que la période absolue T_a est constante, lorsque la direction du courant et celle du train de vague sont de sens opposées, w_r augmente au fur et à mesure que la vitesse du courant est plus élevée. Lorsque la direction du courant et celle du train de vagues sont de même sens, w_r diminue au fur et à mesure que la vitesse du courant est plus élevée.

A partir de la relation de dispersion en pleine mer (3.18), l'équation (4.2) nous permet de calculer la nouvelle valeur de k au point x :

$$k(U) = \frac{1}{g}\left(\frac{2p}{T_a} - \|k_\infty\| \|U\| cos(a)\right)^2 \qquad (4.3)$$

soit :

$$L(U) = \frac{2pg}{\left(\frac{2p}{T_a} - \|k_\infty\| \|U\| cos(a)\right)^2} \qquad (4.4)$$

où a est l'angle que forment les vecteurs nombre d'onde et courant, et où $\| U \|$ est la vitesse du courant au point x qui est exprimée en m/s.

Lorsque la direction du courant et celle du train de vagues sont de sens opposés, la longueur d'onde L diminue au fur et à mesure que la vitesse du courant est plus élevée. Lorsque la direction du courant et celle de la vague sont de même sens, la longueur d'onde L augmente au fur et à mesure que la vitesse du courant est plus élevée.

Il nous reste, maintenant, à calculer l'évolution de la hauteur de la vague en fonction de la variation de la longueur d'onde. En l'absence de dissipation d'énergie, il y a conservation de l'énergie mécanique de la vague. Donc, si la longueur d'onde de la vague diminue, sa hauteur va augmenter. Réciproquement, si sa longueur d'onde augmente, sa hauteur va diminuer.

L'énergie mécanique est égale à la somme des énergies potentielle et cinétique. L'énergie potentielle est l'énergie nécessaire pour déformer la surface de la mer et lui donner l'aspect ondulé tandis que l'énergie cinétique est celle qui est nécessaire pour communiquer aux particules leurs mouvements ondulatoires [Lac65] [Kin65] :

$$E = E_p + E_c = \int_0^L \frac{\rho g}{2} h^2 dx + \frac{\rho}{2} \int_{-h}^{h} \int_0^L \left(u^2 + w^2\right) dx dz = \frac{1}{16} \rho g H^2 L + \frac{1}{16} \rho g H^2 L$$

soit :

$$E = \frac{1}{8} \rho g H^2 L \qquad (4.5)$$

Le principe de conservation de l'énergie mécanique de la vague (4.5) permet d'étudier l'évolution des hauteurs en fonction de la variation des distances parcourues par la vague :

$$H_2^2 L_2 = H_1^2 L_1$$

où H_2 est la nouvelle hauteur ; soit :

$$H_2 = H_1 \sqrt{\frac{L_1}{L_2}} \qquad (4.6)$$

Sachant (4.4), nous pourrions déduire de (4.6) comment la hauteur de la vague varie en fonction de la vitesse du courant U. Seulement, si l'équation (4.5) est directement applicable lorsque la profondeur diminue, ce n'est plus le cas en présence de courant. Il faut tenir compte, en effet, que la pulsation intrinsèque varie pour rendre nos calculs significatifs. Nous utilisons, alors, le principe de conservation de l'action E/w_r de la vague établi par Bretherthon et Garett [BG69]. En l'absence de dissipation d'énergie, sous l'hypothèse d'un écoulement stationnaire, Jonsson [Jon90] a montré que l'équation de conservation de l'action s'écrivait :

$$\frac{\partial}{\partial x}\left(\frac{E\left(C_g + U\right)}{w_r}\right) = 0 \qquad (4.7)$$

où C_g est la vitesse de groupe de la vague et E est l'énergie mécanique de la vague.

La vitesse de groupe de la vague est [Lac65] [kin65] : $C_g = \frac{1}{2}\frac{w_r}{k}\left(1+\frac{2kh}{\sinh(2kh)}\right)$

avec $\frac{2kh}{\sinh(2kh)} \to 0$ en pleine mer et $\to 1$ près du rivage.

Si on pose $n = \frac{2kh}{\sinh(2kh)}$, en remplaçant les valeurs de C_g et de E dans (4.7), on obtient comme nouvelle hauteur de H_2 :

$$H_2 = H_1 \sqrt{\frac{\frac{L_1^2}{4p}(1+n_1) + \frac{L_1 U_1}{w_{r_1}}}{\frac{L_2^2}{4p}(1+n_2) + \frac{L_2 U_2}{w_{r_2}}}} \qquad (4.8)$$

L'océanographe J. Wolf a confronté les données théoriques relatives à l'étude des effets des courants sur les vagues avec des données expérimentales. Dans [Wol99], il réaffirme sa confiance dans l'effet Doppler et dans la loi de Descartes pour étudier numériquement les interactions vague-courant. Sur la figure 17, nous pouvons constater que le front de la vague s'oriente progressivement par rapport à l'axe de la direction contraire du courant de marée.

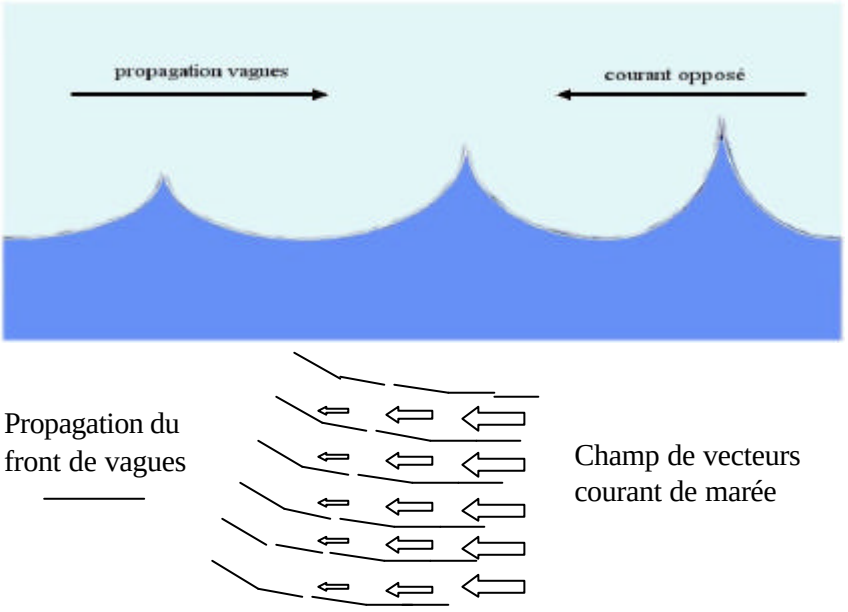


Fig. 17 : Propagation en présence d'un courant contraire de plus en plus fort

Pour exploiter les champs des vecteurs courants de marées des stations de cartographie embarquées à bord des navires (fig. 7), nous effectuons une interpolation bilinéaire des vecteurs courants de marées les plus proches du point x. Puis, afin de prendre en compte les variations du courant, nous tenons compte des variations continues du nombre d'onde k tout le long de la zone de simulation. Le calcul de la phase m devient :

$$m = -w_r t + \int_0^{x_0} k(U)\,dx$$

avec $k(U)$ la valeur de k au point x dont le vecteur courant est U.

Pour être appliquée directement dans un modèle numérique, il suffit de discrétiser, par exemple par la méthode de Riemman, cette intégrale.

En pleine mer

En pleine mer, pour une onde de surface de vecteur nombre d'onde k, notre modèle est une extension du modèle défini par Gonzato [Gon99]. Il permet de déterminer, en un point initial $X_0 = (x_0,y_0)$ inscrit dans le plan de la mer et une hauteur initiale z_0, sa nouvelle position $X = (x,y)$ dans le plan de la mer et sa hauteur z :

$$\begin{cases} X = X_0 + \frac{k}{\|k\|}\frac{H}{2}\sin(m) \\ z = z_0 - \frac{H}{2}\cos(m) \\ m = -w_r(U_0)t + \sum_{0}^{x_0} k(U)Dx \\ k(U) = \frac{w_r^2(U)}{g} \\ w_r(U) = \frac{2p}{T_a} - \|k_\infty\|\|U\|\cos(a) \end{cases} \tag{4.9}$$

Le modèle proposé, suivant la valeur de la surface libre, est général à la vague de Gerstner et à la forme linéaire des équations de Navier-Stokes. **Il ne dépend logiquement que du courant**. U_0 est la vitesse du courant au point (x_0,y_0) et U est la vitesse du courant en un point (x,y).

Près des côtes

Au bord du rivage, nous devons étudier simultanément les effets de la profondeur et du courant sur les vagues. Pour prendre en compte les effets du courant sur les vagues, nous utilisons l'effet Doppler. L'équation (4.3) nous permet de calculer la nouvelle valeur de la pulsation intrinsèque w_r. À partir de la relation de dispersion (3.17) et de l'approximation proposée par Fournier-Reeves (3.30), nous pouvons calculer la nouvelle valeur du nombre d'ondes k :

Comme , $w_r^2 = gk\tanh(kh)$, on peut écrire que :

$$gk\tanh(kh) \approx gk\sqrt{\tanh(k_\infty h)} = \left(\frac{2p}{T_a} - \|k_\infty\|\|U\|\cos(a)\right)^2$$

soit au point x :

$$k(h,U) = \frac{1}{g\sqrt{\tanh(k_\infty h)}}\left(\frac{2p}{T_a} - \|k_\infty\|\|U\|\cos(a)\right)^2 \tag{4.10}$$

Près des côtes, pour une onde de surface de vecteur nombre d'onde k, notre modèle permet de déterminer, en un point initial $X_0 = (x_0,y_0)$ inscrit dans le plan de la mer et une hauteur initiale z_0, sa nouvelle position $X = (x,y)$ dans le plan de la mer et sa hauteur z :

$$\begin{cases} X = X_0 + \frac{k}{\|k\|}\left(\frac{H}{2}t'_b S_x \sin(m) + \frac{H}{2}t_b S_z \cos(m)\right) \\ z = z_0 - \frac{H}{2}t'_b S_z \cos(m) + \frac{H}{2}t_b S_x \sin(m) \\ t_b = |\sin(g)| e^{-0.1 h_0}, \; t'_b = \sqrt{1 - t_b^2} \\ S_x = \frac{1}{1 - e^{-0.11 h_0}}, \; S_z = S_x\left(1 - e^{-0.09 h_0}\right) \\ S_x \leq \frac{1}{\frac{H}{2}k\cos(b)} \\ m = -w_r(U_0)t + \sum_0^{x_0} k(U,h)Dx \\ k(U,h) = \frac{w_r^2(U)}{g\sqrt{\tanh(k_\infty h)}} \\ w_r(U) = \frac{2p}{T_a} - \|k_\infty\|\|U\|\cos(a) \end{cases} \qquad (4.11)$$

Le modèle proposé dépend logiquement de la profondeur et de la vitesse du courant. U_0 et h_0 sont la vitesse du courant et la profondeur au point (x_0,y_0) tandis que U et h sont la vitesse du courant et la profondeur au point (x,y). S_x et S_z ont été fixés expérimentalement par Fournier-Reeves mais nous pouvons également choisir des valeurs plus théoriques : (3.22) pour les vagues de faible amplitude ou stokes d'ordre 2 pour les vagues d'amplitude finie.

La valeur du nombre d'onde est calculée à partir de l'approximation proposée par Fournier et Reeves (3.30). Pour le calcul de la phase, il est tenu compte des variations continues du courant et de la profondeur tout le long de la surface à modéliser.

Au cours de la simulation, la valeur de S_x est toujours calculée en premier au cas où il faudrait luit affecter la valeur limite supérieure de $\frac{1}{\frac{H}{2}k\cos(b)}$. La valeur de S_z est ensuite déduite de la valeur de S_x.

Au final, en pleine mer et près des côtes, seul le modèle (4.11) est exploité comme, lorsque la profondeur augmente, on obtient le modèle (4.9).

4.2.2 L'algorithme de lancer de vague à partir d'un arc de cercle

L'algorithme du lancer de vague a été présenté dans le paragraphe 3.3.2.3. Pour l'appliquer, nous devons définir un front de vague initial.

Dans [TS87], le front initial d'un train de vagues est défini par un seul vecteur k. Dans [Gon99], le front initial d'un train de vagues est une ligne droite ; en chaque point de la droite, il n'est pas utile de définir la direction de propagation comme elle est implicitement orthogonale à la droite. Cette approche est en pratique trop approximative, même si on considère que l'aire du vent générateur est située à l'infini. Une onde ne progresse pas en ligne droite ; dans notre cas, elle progresse suivant un axe que nous avons défini comme étant la droite passant par le centre de la zone de *fetch* et par le point (p_i sur la figure 18) appartenant au front de vague initial. Cela engendre des erreurs sur la position du front de vague, d'autant plus évidentes que la zone parcourue par le navire est large (fig. 18).

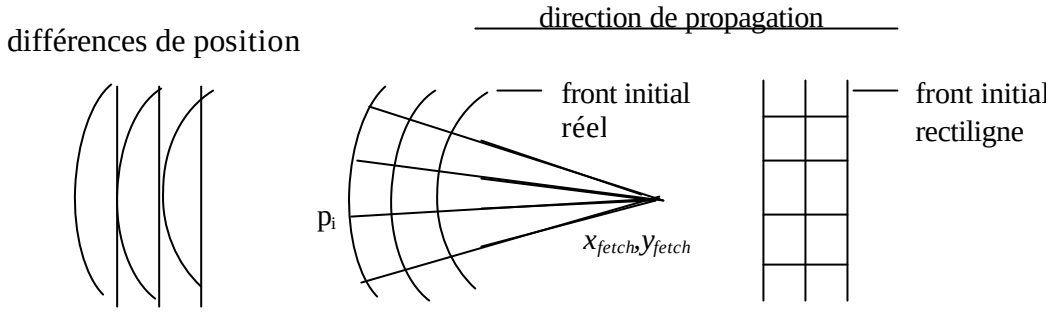
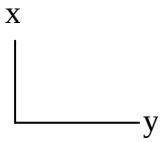


Fig. 18 : Propagation du front de vague

Nous définissons le front initial d'un train de vagues comme étant un arc de cercle. La direction initiale en chaque point de l'arc de cercle est définie par l'axe passant par le point et le centre de la zone de *fetch* ; sur cet axe, le sens de propagation traduit l'éloignement progressif de la zone de *fetch.* Nous délimitons le nombre initial de rayons à lancer en augmentant graduellement la valeur de q en partant de la borne inférieure q_{min} jusqu'à la borne supérieure q_{max} :



$$\begin{cases} x_0 = x_{fetch} + d\sin(q) \\ y_0 = y_{fetch} + d\cos(q) \\ q_{min} \le q \le q_{max} \end{cases} \qquad (4.12)$$

où la distance d, q_{min} et q_{max} sont des paramètres d'entrée du système (fig. 19) et q la direction du vecteur nombre d'onde k au point x_0, y_0.

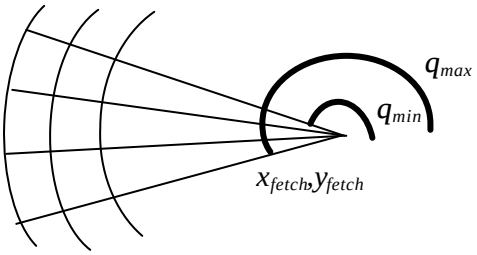


Fig. 19 : Définition du front initial de vague

Parce que la longueur d'onde et la période peuvent varier simultanément en présence de courant près des côtes, nous utilisons la loi de Descartes pour déterminer les changements successifs de la direction de propagation des vagues :

$$\frac{\sin(a_1)}{C_1}=\frac{\sin(a_2)}{C_2} \tag{4.13}$$

La décomposition en trois dimensions du modèle (4.11) s'écrit :

$$\begin{cases} x=x_0+\frac{k_x}{\|k\|}(A\sin(m)+A'\cos(m)) \\ y=y_0+\frac{k_y}{\|k\|}(A\sin(m)+A'\cos(m)) \\ z=z_0-B\cos(m)+B'\sin(m) \end{cases}$$

avec : $A=\frac{H}{2}t'_b S_x,\ A'=\frac{H}{2}t_b S_z,\ B=\frac{H}{2}t'_b S_z,\ B'=\frac{H}{2}t_b S_x$

Afin de l'appliquer ultérieurement en temps réel, à l'issue de l'algorithme de lancer de vague, il est mémorisé en chaque point (x_0,y_0) d'une grille 2D :

. l'intensité k du vecteur nombre d'onde et sa direction de propagation q
. les quatre coefficients de l'ellipse A,A',B,B'
. la pulsation relative w_r
. la phase non temporelle

L'ensemble des grilles 2D recouvre la zone de navigation.

4.3 Somme classique des vagues et simulation spectrale

4.3.1 La somme classique des vagues

Jusqu'à maintenant nous avons étudié un seul train de vagues de gravité. Nous souhaiterions simuler, maintenant, la forme et la propagation d'un groupe de trains de vagues de gravité formant **une houle,** laquelle est néanmoins constituée de moins de trains de vagues que la mer du vent. Pour cela, nous calculons individuellement la forme et la propagation de chaque train de vagues de gravité à l'aide de l'algorithme de lancer de vague défini à partir d'un arc de cercle. Puis, nous effectuons en temps réel la somme des vagues. En pleine mer, sans tenir compte du courant de marée, la nouvelle position dans le plan de la mer $D = (x,y)$ et la hauteur h au point initial $X_0 = (x_0,y_0)$ inscrit dans le plan de la mer de hauteur z_0 sont :

$$\begin{cases} D(X_0,t) = X_0 + \sum_k \left(\frac{k}{\|k\|}\right) a_k \sin(k.X_0 - w_r t) \\ h(X_0,t) = z_0 - \sum_k a_k \cos(k.X_0 - w_r t) \end{cases} \tag{4.14}$$

L'approche est exacte si on néglige les dissipations d'énergie dues aux effets non linéaires entre les vagues et si on adopte le principe de linéarisation qui permet de superposer tous les mouvements.

L'intérêt de cette approche est de pouvoir simuler simultanément la forme et la propagation de plusieurs trains de vagues (**une houle**) en tenant compte des effets de la profondeur et du courant de marée. Lors de l'algorithme de lancer de vague, nous appliquons le modèle (3.31) pour chaque train de vagues. Au cours des pré-calculs, la forme et la propagation de chacune des ondes de surface sont mémorisées au sein de grilles 2D qui recouvrent l'ensemble de la zone de navigation. Lors d'une session de simulation, chaque onde de surface est ensuite reconstituée en 3D avant que la somme ne soit effectuée. Toutefois, l'exploitation temps réel de cette approche est conditionnelle à une augmentation de la puissance de calcul par rapport à ce qui est actuellement disponible à un coût raisonnable. De plus, toute augmentation des ressources de calcul peut être consommée en augmentant le nombre de trains de vagues à traiter.

Nos expérimentations sont présentées dans le chapitre suivant qui est prévu à cet effet. Toutefois, l'approche qui consiste à effectuer une somme ordinaire des vagues nous a conduit dans une impasse due à un manque de ressources de calcul. Nous avons donc procédé à une répartition des traitements, par une approche traditionnelle, dont nous décrivons, très sommairement dans ce paragraphe, l'expérimentation s'y rapportant.

Performances

Sur un PC doté d'un processeur de calcul Pentium III 865 Mhz et d'une carte graphique grand public NVIDIA Geforce 4 Ti 4600, nous obtenons un taux approximatif de 10 images par seconde lorsque nous étudions la forme et la propagation d'un seul train de vagues. Notre résolution spatiale est d'un point tous les mètres ; nos structures de données sont exclusivement des tableaux.

Nous constatons que les performances se dégradent linéairement au fur et à mesure que le nombre de trains de vagues augmente. Lorsque nous effectuons la somme de trois trains de vagues, le taux d'images passe à 3 images par seconde. Le seul point positif de cette expérimentation est que l'utilisation de trois trains de vagues suffit à représenter de manière réaliste une houle peu formée (un groupe de trains de vagues) qui se propage vers la côte. En présence de dix trains de vagues, le taux d'images est à peine d'une image par seconde.

Pour mener à bien cette expérimentation, nous avons positionné le navire en pleine mer afin de ne pas perturber les résultats par l'affichage du littoral. Pour le rendu, nous n'utilisons pas de modèle d'éclairage : pour distinguer correctement la forme géométrique des vagues, la couleur associée à un point du maillage est choisie en fonction de la hauteur du point. Nous choisissons un cône de vision dont le champ horizontal est de 60 °. Au-delà d'une portée de 5 km, nous n'affichons plus la mer.

Pour pallier le manque de ressources de calcul, nous avons alors renouvelé l'expérimentation mais cette fois-ci sur un PC bi-processeurs doté de deux processeurs de calcul Pentium III 865 MHz et

de la même carte graphique grand public NVIDIA Geforce 4 Ti 4600. Nous répartissons alors le calcul des surfaces de mer entre les deux processeurs de calcul. Seulement, comme à un processus est associé un seul contexte graphique, un seul des deux processus est en charge de l'affichage.

Pour mener à bien cette nouvelle expérimentation, avec les mêmes paramètres que précédemment, nous nous sommes appuyés sur une programmation multi-thread. Les deux processus communiquent par l'intermédiaire d'une boîte à lettres qui est une structure de données de type file. La répartition des traitements est choisie de manière à optimiser le temps d'occupation de chaque processeur. Lorsque le premier processus calcule une surface de mer, il l'affiche instantanément. Lorsque le deuxième processus a terminé de calculer une surface de mer, il informe le premier au moyen de la boîte aux lettres qu'un affichage est possible.

Lors de cette deuxième expérimentation, nous avons constaté que nous dégradions systématiquement les performances. À titre d'exemple, le taux d'images pour afficher un seul train de vagues a été de 3 images par seconde. La raison à cela, qui est rédhibitoire, est le temps d'accès à la mémoire. Alors que, dans le premier cas, l'affichage était effectué « à la volée », dans le deuxième cas, il se crée un transfert massif de données entre les deux processus qui engendre des points d'étranglement et dégrade les performances.

L'étape suivante consisterait à recommencer l'expérimentation sur une machine parallèle mais nous nous éloignerons d'une configuration matérielle grand public que privilégie un lycée maritime. Le manque de contrôle du temps de la construction d'une image nous a amené à approfondir le modèle de J. Tessendorf.

4.3.2 Extensions du modèle de Tessendorf

Nous avons vu dans le paragraphe 3.3.3.2 que le modèle de J. Tessendorf était un modèle statistique, pour lequel la hauteur des vagues dépendait d'un spectre initial mais également de deux nombres aléatoires x_r et x_i.

Pour implémenter ce modèle, nous avons utilisé une loi normale centrée réduite, de moyenne égale à 0 et d'écart type égale à 1. La taille de notre *patch FFT* est de 256 m sur 256 m avec une résolution spatiale d'un point tous les mètres. Cette configuration n'est pas celle qu'utilise J. Tessendorf ; n'ayant pas à respecter la contrainte de temps réel, il utilise un *patch FFT* de 2048 m sur 2048 m avec une résolution spatiale d'un point tous les 3 cm.

Avec notre résolution spatiale, nous constatons que, dans des conditions de vent identiques, où la direction du vent et sa vitesse sont constantes (31 km/h pour les résultats présentés sur la figure 20), les surfaces de mer obtenues sont très différentes d'une exécution à l'autre, correspondant ici aux *états de mer* 4 et 5 de l'échelle de Beaufort de la figure 20 pour lesquels les vitesses du vent sont théoriquement comprises entre 20 et 28 km/h et entre 29 et 31 km/h. De plus, nous constatons que les surfaces de mer sont parfois peu représentatives de la mer du vent.

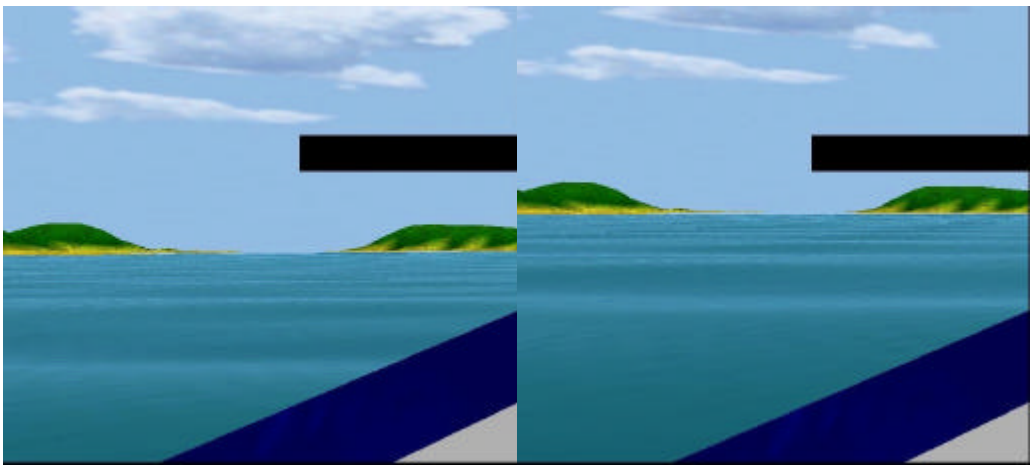


Fig. 20 : Etats de mer : 4 (à gauche), 5 (à droite)

L'enseignant d'un lycée maritime dispose de peu de temps pour configurer les paramètres du système dans le but de préparer un exercice type (correspondant à un *état de mer* par exemple). Dès lors, la diversité des surfaces de mer, que produit le modèle de J. Tessendorf dans des conditions de vent identiques, nous pose problème. L'objet de ce qui suit est de déterminer sur quelle théorie physique de la vague ce modèle s'appuie et de préciser où interviennent les deux nombres aléatoires x_r et x_i. Pour y parvenir, au lieu d'effectuer une somme d'ondes de surface, nous délimitons l'étude à une seule onde de surface de nombre d'onde k.

Compte-tenu de la réécriture d'une somme de fonctions sinusoïdales en une somme de nombres complexes et de leurs nombres complexes conjugués (3.34), pour comprendre la trajectoire des particules d'eau en présence d'une seule onde de surface de nombre d'onde k, il est nécessaire de faire la somme de l'onde de surface de nombre d'onde k avec l'onde de surface de nombre d'onde $-k$. Pour faciliter la compréhension, nous posons $a_k = x_r\sqrt{Ph(k)}/\sqrt{2}$ et $b_k = x_i\sqrt{Ph(k)}/\sqrt{2}$ et m est la phase.

En vue de profil, pour l'onde de surface de nombre d'onde k, on obtient comme déplacement suivant l'axe x et suivant l'axe z :

$$\begin{cases} x = b_k cos(m) + a_k sin(m) \\ z = a_k cos(m) - b_k sin(m) \end{cases}$$

Pour vérifier que l'on obtient bien une trajectoire circulaire, on applique l'équation du cercle. On obtient :

$$x^2 + z^2 = \left(b_k cos(m) + a_k sin(m)\right)^2 + \left(a_k cos(m) - b_k sin(m)\right)^2 = a_k^2 + b_k^2 = \frac{Ph(k)}{2}\left(x_r^2 + x_i^2\right) \quad (4.15)$$

La trajectoire des particules décrit un cercle dont le rayon est donné par l'équation (4.15).

Une première remarque est que, pour une onde, la phase est généralement définie comme étant $m = kx_0 - w_r t$ plutôt que $m = kx_0 + w_r t$. Une deuxième remarque est que, si la trajectoire des particules décrit bien un cercle, les équations suivant chaque axe ne correspondent à aucune des théories physiques de la vague qui ont été présentées dans le paragraphe 3.3. Les termes des équations suivant chaque axe ont été choisies de manière à ce que le rayon du cercle dépende des deux nombres aléatoires x_r et x_i (4.15).

Nous conseillons de ne générer qu'un seul nombre aléatoire plutôt que deux pour respecter les corrélations existantes entre une vitesse du vent donnée et un *état de mer* quelle que soit la résolution spatiale choisie (voir l'échelle de Beaufort présentée en *annexe A*). L'équation (3.39) deviendrait :

$$\tilde{h}_0(k) = x(1+i)\sqrt{\frac{P_h(k)}{2}}$$

De plus, pour respecter la conformité de l'échelle de Beaufort, nous conseillons de choisir un intervalle de valeurs très faible pour le nombre aléatoire x, par exemple entre –0.5 et +0.5. D'autre part, il n'était pas nécessaire de définir les deux termes a_k et b_k pour traiter l'équation d'un cercle. Seul le terme a_k suffisait compte-tenu de la propriété de parité et de la propriété de conjugaison des nombres complexes.

Pour traiter plusieurs profils de vagues, nous devons dissocier le champ d'amplitudes suivant l'axe z du déplacement D dans le plan de la mer. On peut alors introduire les pentes t_b et t_b' d'une ellipse conformément à notre modèle général (4.11). Si les phases temporelles sont interverties par rapport au modèle initial (3.37), cela est dû à la définition générale de la phase d'une onde qui est plutôt $m = kx - w_r t$ que $m = kx + w_r t$. Le modèle de J. Tessendorf (3.36) devient (4.16) :

$$\begin{cases} D(X_0,t) = X_0 + \frac{1}{2N}\sum_k \left(-i\left(\frac{k}{\|k\|}\right)\tilde{d}_0(k)exp(-iw_r t) + i\left(\frac{-k}{\|-k\|}\right)\tilde{d}_0^*(-k)exp(iw_r t)\right)exp(i\,k.X_0) \\ h(X_0,t) = \frac{1}{2N}\sum_k -\left(\tilde{h}_0(k)exp(-iw_r t) + \tilde{h}_0^*(-k)exp(iw_r t)\right)exp(i\,k.X_0) \end{cases}$$

On vérifie que lorsque :

$\tilde{h}_0(k) = \tilde{d}_0(k) = \frac{P_h(k)}{2}$, on est en présence de la vague de Gerstner (de Airy) (3.25) (3.23)

$$\begin{cases} \tilde{h}_0(k) = \frac{P_h(k)}{2}S_z \\ \tilde{d}_0(k) = \frac{P_h(k)}{2}S_x \end{cases}$$, on est en présence de la vague de faible amplitude (3.22)

$$\begin{cases} \tilde{h}_0(k) = \frac{P_h(k)}{2}t'_b S_z + i\frac{P_h(k)}{2}t_b S_x \\ \tilde{d}_0(k) = \frac{P_h(k)}{2}t'_b S_x + i\frac{P_h(k)}{2}t_b S_z \end{cases}$$, on est en présence d'une vague de Biesel(4.11)

Pour une vague de Biesel, la trajectoire des particules est une ellipse dont le grand axe s'oriente progressivement parallèlement à la pente du sol (fig. 21).

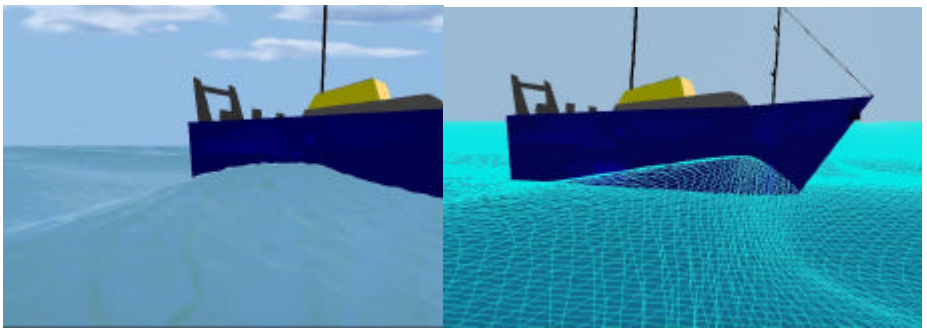


Fig. 21 : Somme de vagues de Biesel avec les FFT

Compte-tenu de la réécriture d'une somme de fonctions sinusoïdales en une somme de nombres complexes et de leurs nombres complexes conjugués (3.34), pour comprendre la trajectoire des particules d'eau en présence d'une seule onde de surface de nombre d'onde k, il est nécessaire de faire la somme de l'onde de surface de nombre d'onde k avec l'onde de surface de nombre d'onde $-k$. Pour faciliter la compréhension, nous posons $a_k = x_r\sqrt{P_h(k)}/\sqrt{2}$ et $b_k = x_i\sqrt{P_h(k)}/\sqrt{2}$ et m est la phase.

En vue de profil, pour une vague de nombre d'onde k, on obtient comme déplacement suivant l'axe x et suivant l'axe z :

$$\begin{cases} x = b_k \cos(m) + a_k \sin(m) \\ z = -a_k \cos(m) + b_k \sin(m) \end{cases}$$

Conformément à notre modèle général (4.11), les valeurs de (a_k,b_k) sont :

$$\left(\frac{P_h(k)}{2} t'_b S_z, \frac{P_h(k)}{2} t_b S_x\right) \text{ et } \left(\frac{P_h(k)}{2} t'_b S_x, \frac{P_h(k)}{2} t_b S_z\right)$$

On obtient bien un terme en $-$cosinus et un terme en +sinus suivant l'axe z et un terme en +cosinus et en +sinus suivant l'axe x. La trajectoire des particules est une ellipse (fig 22).

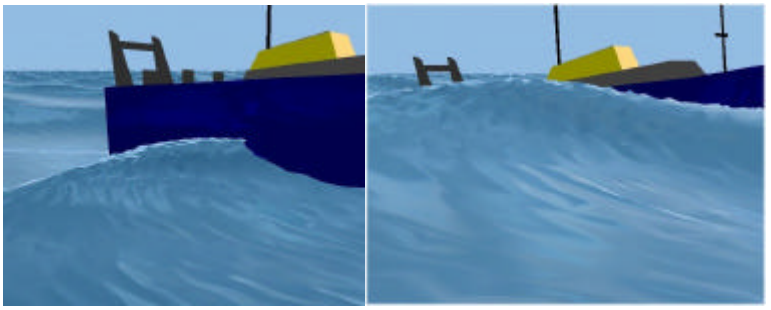


Fig. 22 : Notre modèle de mer du vent

A noter que lorsque la profondeur augmente, le terme b_k®0 (t_b®0), les fonctions $h(X_0, t)$ et $D(X_0, t)$ deviennent paires. On obtient alors la vague de Gerstner.

Pour déterminer l'amplitude des vagues dans le domaine des fréquences, nous utilisons le spectre de Phillips (3.38), mais nous avons également expérimenté les spectres de Pierson-Moskowitz et d'Hasselmann pour constater qu'ils donnaient sensiblement les même surfaces de mer. Une autre approche serait alors d'utiliser, non plus un spectre théorique, mais un spectre expérimental obtenu à partir d'instruments de mesure. Un spectre expérimental correspond à une condition de vent donnée. La somme des vagues est dans un premier temps décomposée en fonctions harmoniques simples. Puis, lors de la simulation, la reconstruction est opérée.

Pour couvrir un large spectre de vagues océaniques, nous choisissons N=128, soit des *patchs* de 256 m sur 256 m. La plus grande longueur d'onde que nous pouvons introduire est donc de 128 m dans toutes les directions possibles du plan de la mer. Puis, on calcule les hauteurs suivant l'axe *z*, et les déplacements dans le plan suivant l'axe *x* et suivant l'axe *y*.

Pour introduire les effets du courant et de la profondeur, nous proposons l'approche suivante. Nous sommes exactement dans la situation inverse où nous étions dans le paragraphe 4.2.1 : $k_¥$ était donné ; nous calculions w_r (qui variait à cause du courant), puis k (qui variait à cause de la profondeur). En simulation spectrale, les valeurs des nombres d'onde k sont fixes ; le domaine des fréquences est le domaine des points qui constitue une grille 2D ; le pas qui sépare les lignes et les colonnes de la grille est fixe. Nous considérons alors que les nombres d'onde k représentent nos nombres d'onde après transformation due au courant et/ou à la profondeur ; le seul paramètre sur lequel nous pouvons interagir est le champ des amplitudes. k étant donné, nous recherchons tout d'abord la valeur du nombre d'onde $k_¥$ initial à partir de la relation de dispersion (3.17) (3.18) et de l'effet Doppler (4.1) :

$$\begin{cases} k_\infty = k\tanh(kh) & \text{près des côtes, sans courant} \quad (4.17) \\ k_\infty = \left(\dfrac{2p}{T_a} - \sqrt{gk}\right) / \|U\| cos(a) & \text{en pleine mer en présence de courant} \\ k_\infty = \left(\dfrac{2p}{T_a} - \sqrt{gk\tanh(kh)}\right) / \|U\| cos(a) & \text{près des côtes en présence de courant} \end{cases}$$

Connaissant $k_¥$., le nombre d'onde en pleine mer sans courant, nous appliquons le spectre de Phillips pour déterminer la hauteur initiale de la vague $Ph(k_¥)$. Ensuite, à partir des principes de conservation de l'énergie/action, nous finissons par calculer la nouvelle amplitude $Ph(k)$ associée au nombre d'onde k (les étapes successives sont représentées sur la fig. 23) ; dans le cas le plus simple, à partir du principe de conservation de l'énergie :

$$Ph(k) = Ph(k_\infty)\sqrt{\frac{L_\infty}{L}} \qquad (4.18)$$

k ——— $k_{¥}$ ——— $Ph(k_{¥})$ ——— $Ph(k)$

Fig. 23 : Étapes successives pour calculer le nouveau champ d'amplitudes

Enfin, la FFT inverse (4.15) permet de reconstruire les surfaces de mer en utilisant le modèle le plus complet (4.11).

Nous ne prenons pas en considération le changement de propagation des vagues durant la simulation spectrale, parce que, par définition, cela est impossible. C'est le même *patch FFT* qui est dupliqué sur toute la surface de l'océan.

L'effet le plus visuel du au courant ou à la profondeur est la variation du champ des hauteurs et le changement de la forme des vagues due aux modifications des trajectoires des particules d'eau.

4.4 Les vagues de courtes longueurs d'onde

Nous appelons vagues de courtes longueurs d'onde les vagues capillaires (longueurs d'onde < 1.74 cm) et ce qui est couramment appelé, en termes maritimes, le clapotis (petites vagues qui se superposent). Ce type de vagues est souvent introduit par une approche fractale. Des fonctions bruits sont alors utilisées [Gon99]. Nous choisissons plutôt l'approche FFT qui est beaucoup plus représentative des surfaces de mer.

Nous choisissons un spectre théorique simple (4.19), dépendant de la vitesse du vent. Plus la vitesse du vent est élevée, plus le nombre de vagues de courtes longueurs d'onde est élevé et plus l'allure de la surface de la mer semble hachée. En présence d'une faible brise, la surface est plus régulière.

$$Cap(k) = A\frac{1}{k}ln(1+\|W\|)\left|\frac{k}{\|k\|}\cdot\frac{W}{\|W\|}\right| \qquad (4.19)$$

où $||W||$ est la vitesse du vent, et où le produit scalaire porte sur les vecteurs direction du nombre d'onde et direction du vent. Il est choisi comme valeur 0 lorsque le résultat du produit scalaire est négatif ou nul parce que ce type de vagues suit toujours la direction du vent.

En fonction de la magnitude de chaque nombre d'onde k, nous appliquons le spectre de Phillips (3.38) ou le spectre (4.19). La valeur de la constante A est à fixer expérimentalement ; nous avons choisi $A = 0.1$ pour réaliser l'illustration de la figure 24, où la résolution spatiale est d'un point tous les 1 cm. Lorsque la longueur d'onde diminue, la hauteur diminue également grâce au terme $1/k$; elle augmente en fonction de la vitesse du vent mais tend vers une limite grâce à la fonction logarithmique. Enfin lorsque la longueur d'onde augmente, on utilise alors le spectre (4.19) ; de la sorte, le cas $Cap(k) \to ¥$ lorsque $1/k \to 0$ ne peut pas se produire.

Fig. 24 : Les vagues de courtes longueurs d'onde

A noter que, pour les vagues capillaires (< 1.74 cm), la tension de surface de l'eau intervient dans la relation de dispersion. Un terme additionnel est ajouté [Kin65] [Lac65] :

$$w_r^2 = gk + \frac{g\,k^3}{r}$$

Soit :

$$w_r^2 = gk\left(1 + \frac{g\,k^2}{r\,g}\right)$$

où γ est la tension de surface pour la surface de contact air-eau à 20° (T_1 = 74 10^{-5} Ncm^{-1} = 74 10^{-3} Nm^{-1}), r est la masse volumique de l'eau à la surface de la mer (r = 1 gcm^{-3} = 1000 kgm^{-3}), g est la gravité terrestre (g = 9,81 ms^{-2}).

D'où :

$$w_r^2 = gk\left(1 + \frac{74\,10^{-6}k^2}{g}\right) \qquad (4.20)$$

4.5 Résumé du chapitre

Dans ce chapitre nous avons proposé :

- un modèle unique pour simuler en temps réel la forme et la propagation de la houle (un seul train régulier de vagues) depuis la pleine mer jusqu'au rivage qui prend en compte les effets du courant de marée et de la profondeur. L'algorithme de lancer de vague est de type lancer de vague à partir d'un arc de cercle.

- une extension du modèle de J. Tessendorf pour simuler en temps réel la génération des vagues de gravité sous l'effet du vent qui prend en compte ponctuellement les effets de la profondeur et du courant de marée.

- un spectre théorique pour les vagues de courtes longueurs d'onde qui peut être utilisé en temps réel conjointement avec les modèles de houle et de mer du vent.

On ne peut pas utiliser simultanément les modèles de houle et de mer du vent ; il n'existe pas de continuité entre ces deux modèles parce que nous n'étudions pas la propagation des trains de vagues avec le modèle de mer du vent. Lors de la définition d'un exercice, il est alors proposé à l'enseignant de choisir l'un ou l'autre de ces deux modèles.

Chapitre 5

Simulation temps réel des océans

5.1 Introduction

Les premiers paragraphes de ce chapitre ont été introduits pour faciliter la réalisation d'applications industrielles. Nous pensions qu'il était nécessaire de proposer une démarche générale pour parvenir à simuler en temps réel des scènes océanes.

Dans le deuxième paragraphe, nous commençons par énoncer les contraintes que nous devons satisfaire. Pour y parvenir, nous avons choisi une structure de données multigrille à échelle fixe que nous décrivons dès le début de ce chapitre. Puis, tout au long du chapitre, nous expliquons les traitements qui s'y rapportent.

Pour construire une image, nous nous appuyons sur un ensemble de données initiales. Tout traitement des données initiales qui peut être effectué préalablement à la simulation permet ensuite de diminuer le temps de la construction d'une image. L'ensemble des pré-traitements des données initiales est présenté dans le quatrième paragraphe.

Le moyen le plus adapté pour décrire la démarche générale que nous proposons est de présenter l'architecture logicielle de notre système. L'enchaînement successif des traitements portant sur les données y est clairement illustré.

La construction temps réel d'une image amène à contrôler simultanément le temps de calcul et le temps d'affichage de l'image. Certains éléments de la scène 3D sont en mouvements et, à chaque pas de simulation, nous devons calculer leurs déplacements (particules d'eau, notre navire, les autres navires, les bancs de poissons, ...), d'autres ne le sont pas (côte, balises, ...). C'est l'appel aux primitives de la bibliothèque graphique [OGL96] qui permet de réaliser l'affichage des éléments de la scène 3D. Le premier procédé traditionnel pour diminuer simultanément les temps de calcul et d'affichage est d'exclure les éléments de la scène 3D qui ne sont pas inclus dans le cône de vision afin de ne pas traiter plus d'éléments qu'il n'en faut. Concernant les éléments inclus dans le cône de vision, nous procédons à des calculs parallèles dès qu'il nous est possible de le faire. De plus, nous utilisons plusieurs simplifications pour diminuer le nombre de polygones constitutifs des éléments à afficher. Certaines de ces simplifications sont conventionnelles en synthèse d'images comme l'affichage par niveaux de détails des surfaces de mer qui est présenté dans le septième paragraphe. D'autres simplifications sont moins conventionnelles comme l'utilisation d'un plan texturé sous la mer pour masquer les craquelures entre les surfaces de mer qui présentent des résolutions spatiales différentes.

5.2 Enoncé des contraintes

Pour maintenir une cinématique proche de celle du temps réel, nous devons rechercher un taux d'images qui permette d'avoir au minimum 20 images par seconde ; c'est notre contrainte majeure (contrainte 1). Elle est dépendante de la configuration matérielle. À cet effet, les capacités des processeurs de calcul et des cartes graphiques ont augmenté régulièrement ces dernières années, mais encore faut-il les utiliser à bon escient pour maintenir la fluidité de l'animation.

Avant d'énoncer les autres contraintes, qui ne sont pas sans être en relation avec notre contrainte majeure, il est une contrainte conventionnelle des animations 3D que nous n'avons pas. Au cours d'une session de simulation, le navire est susceptible de se mouvoir à n'importe quel endroit du moment qu'il ne s'échoue pas. Si cela arrive, la simulation s'arrête, ce qui nous prémunit de l'effet indésirable de voir le navire traverser la côte. La détection des collisions entre le navire et le paysage est un traitement que nous n'avons pas à effectuer. Il ne sera donc pas abordé dans ce chapitre.

La deuxième contrainte résulte de la liberté de mouvement d'un navire et de la configuration matérielle grand public que nous ciblons. Avec une configuration matérielle de type PC, nous ne pouvons pas charger en mémoire l'ensemble des données du littoral (données d'élévation, arbres, maisons, bâtiments, …) sans saturer la capacité de la mémoire du processeur de calcul, ce qui aurait comme conséquence immédiate d'altérer les performances. Notre deuxième contrainte (contrainte 2) porte sur la délimitation des données à traiter pour ne pas saturer la mémoire du processeur de calcul.

Enfin, dans [Tes01-1], J. Tessendorf a montré que seule une résolution spatiale inférieure au centimètre permettait d'obtenir une surface de mer représentative (fig. 25).

Fig. 25 : Surfaces de mer avec un point tous les 100 cm, 10 cm, 1 cm

La démonstration est encore plus pertinente lorsque nous ajoutons les reflets du soleil sur la mer. Seule une résolution spatiale inférieure au centimètre permet de représenter les points brillants que nous pouvons observer lorsque le soleil se reflète sur la mer (fig. 26). La granularité extrême d'une surface de mer, plus importante que la granularité d'une surface de terrain, constitue notre troisième contrainte (contrainte 3).

Fig. 26 : Images de synthèse du film Titanic

5.3 Notre structure de données

En simulation d'entraînement maritime, la caméra, que nous commençons par définir, est supposée être positionnée sur la passerelle du navire, recréant le champ de vision de l'homme à la barre. À cet effet, nous définissons tout d'abord un cône de vision dont le champ horizontal est de 60 degrés. En deçà de cette valeur, nous constatons une déformation de la coque du navire. Au-delà de cette valeur, les principaux fabricants européens de simulateurs [Trans] [Kongs] commercialisent une configuration matérielle constituée de plusieurs écrans pour recréer un champ de vision humaine de 180 degrés (fig. 2). Nous positionnons à 1 m le premier plan de coupe du cône de vision (plan de coupe « near » sur la figure 27) tandis que la portée maximale du cône de vision (plan de coupe « far » sur la figure 27) dépend logiquement des conditions de visibilité. Pour effectuer nos tests, nous avons choisi une distance arbitraire de 10 km. Au-delà de cette distance, il n'est plus nécessaire d'afficher le paysage comme il n'est plus visible. Le paysage est subdivisé en éléments de surface d'égales dimensions, ce qui permet d'accélérer les calculs de visibilité.

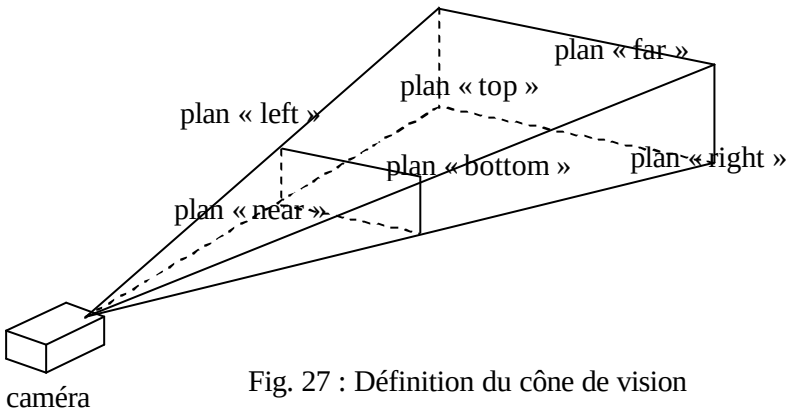


Fig. 27 : Définition du cône de vision

Beaucoup d'articles traitent du rendu de terrains en temps réel [Blo00], [Duc97], [Eva97], [Hec94], [Hop98], [Lin96], [Ögr00], [Paj98], [Rab97], [Red99], ... L'examen de ces travaux nous a conduit, dans un premier temps, à retenir une structure de données multigrille puis, dans un deuxième temps, après expérimentations, à opter pour une structure de données multigrille à échelle fixe.

Tout d'abord, pour satisfaire la contrainte 2, portant sur la répartition des données à traiter, nous procédons à une subdivision planaire du paysage en tuiles d'égales dimensions que nous appelons **découpage de premier niveau** (ou **grille de premier niveau**). Une tuile est une surface carrée dont la dimension est à fixer expérimentalement de telle manière que, lorsque le navire se déplace, le volume des données chargées en mémoire reste à peu près constant. Cette dimension à fixer doit être un multiple de la taille d'un *patch Lancer de vague* et d'un *patch FFT*. Pour effectuer nos tests, nous avons choisi une dimension de 2,56 km (10x10 *patchs* par tuile). Par la suite, nous ne chargeons en mémoire que les données relatives à une tuile traversée ou incluse dans le cône de vision (fig. 28).

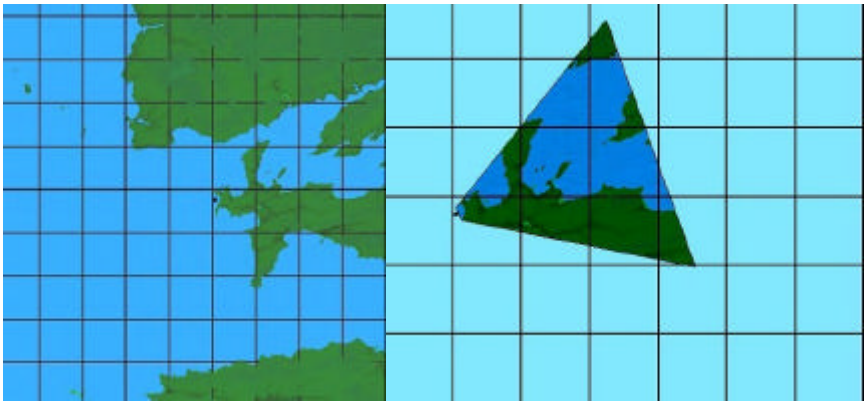


Fig. 28 : Découpage du paysage en tuiles

Parmi les premiers traitements à effectuer, nous devons tout d'abord décider si un point géographique est situé en mer ou sur la côte, à priori quand l'altitude est nulle mais nous verrons que cela nécessite des calculs préalables. Nous procédons à nouveau à une subdivision planaire d'une tuile en éléments de surface d'égales dimensions, que nous appelons **découpage de deuxième niveau (**ou **grille de deuxième niveau).** Au deuxième niveau, nous trouvons les données altimétriques au format Modèle Numérique de Terrains (MNT) [Flo93], ce qui nous permet de dissocier la mer de la côte et de ne pas afficher les maillages qui sont situées à l'extérieur du cône de vision (fig. 28 à droite). Si un point géographique est situé en mer, nous devons connaître sa profondeur ainsi que la pente du fond de la mer, la direction et la force du vent et celles du courant de marées, ... Globalement, nous désignerons ce niveau comme étant le niveau des données initiales (champs de vecteurs courant de marées, champs de vecteurs vent, …).

Nous allons voir que le processus de découpage d'un niveau se poursuit pour l'affichage de la mer (fig. 29), et de la même manière que précédemment en procédant à nouveau à des subdivisions planaires.

Au deuxième niveau, nous disposons de données initiales en des points discrèts ; pour obtenir une valeur en un point quelconque, nous utiliserons systématiquement la méthode d'interpolation bilinéaire, largement couverte en synthèse d'images. Cette méthode est moins précise qu'une méthode d'interpolation bicubique mais elle est beaucoup plus rapide en temps d'exécution.

Pour satisfaire la contrainte 3, portant sur la granularité extrême d'une surface de mer, lorsque des points géographiques consécutifs sont situés en pleine mer, et uniquement dans ce cas, nous procédons à nouveau à une subdivision planaire d'un élément de deuxième niveau en éléments de surface d'égales dimensions, que nous appelons **découpage de troisième niveau (**ou **grille de troisième niveau)**. Au troisième niveau, nous trouvons un un *patch Lancer de vague* ou un *patch FFT* que nous avons défini respectivement dans les paragraphes 3.3.2.3 et 3.3.3.2. Au troisième niveau, notre résolution spatiale est d'un point tous les mètres.

Le dernier niveau est le niveau des textures. Une texture est un tableau, généralement en 2D, de pixels défini par un système de coordonnées (s,t) unitaire [0,1]x[0,1]. Le pixel s'appelle ici un

« texel » ; il représente généralement une couleur, mais peut aussi être une transparence ou une altération de surface (« bump-mapping »). On peut dire que la texture (une image) est « punaisée » sur les sommets d'un polygone : au troisième niveau, avec une résolution spatiale d'un point tous les mètres, nous «punaisons » une texture recouvrant une surface de un mètre carré ; les textures sont dupliquées sur toute la surface de la mer. Pour une texture de 256x256, la résolution spatiale est alors d'environ un point tous les 3.9 mm.

Nous avons ajouté ce dernier niveau pour deux raisons. Concernant la première raison, nous l'avons vu dans le paragraphe 4.4, nous ne pouvons pas traiter correctement les vagues capillaires avec une résolution spatiale d'un point tous les mètres (longueur d'onde < 1.74 cm). La deuxième raison est en rapport avec les bancs de poissons. Si un banc de poissons, un banc de thons par exemple, se tient près de la surface, cela crée des remous. Pour un senneur (*annexe B)*, c'est alors le signal pour entourer le banc de poissons et larguer sa senne (*annexe B*).

Quel que soit le niveau, chaque grille régulière est implémentée par un tableau à deux dimensions.

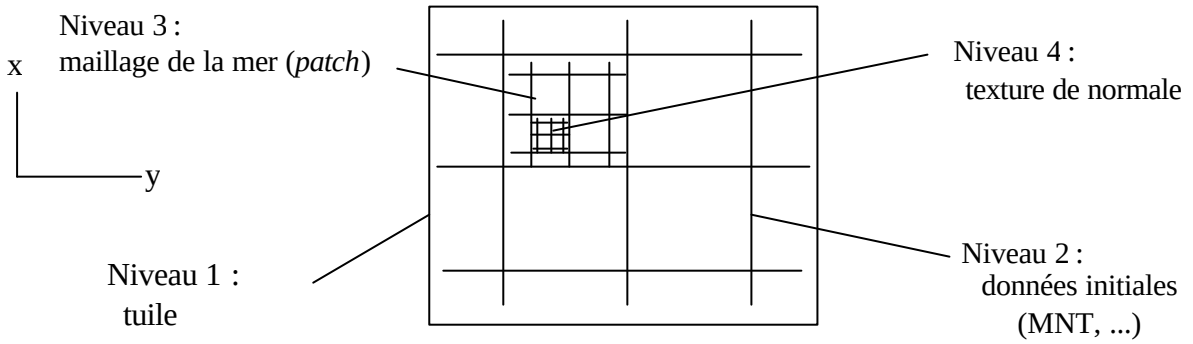


Fig. 29 : Découpage récursif d'un niveau

5.4 Pré-traitements des données initiales

5.4.1 Le littoral et la mer

Nous avons vu précédemment que, lorsque des points géographiques consécutifs étaient situés en pleine mer, nous procédions à nouveau, au deuxième niveau, à une subdivision planaire et nous affichions la mer. Si, pour cela, nous nous basions uniquement sur une valeur nulle des données d'élévation, alors une bonne partie des terres se retrouveraient immergée.

Initialement, il existe en effet beaucoup de points dont l'altitude est nulle dans un fichier MNT. Ce sont tous les points situés géographiquement dans la mer mais ce sont également les points situés dans les terres dont l'altitude est voisine de celle du niveau de la mer. Nous montrons, dans ce paragraphe, comment éliminer ce dernier ensemble de points.

Pour déterminer exactement le littoral et le dissocier de la mer, nous devons donc effectuer un pré-traitement des fichiers MNT. Pour y parvenir, nous définissons autant de graphes non orientés G = <S,A> qu'il existe d'ensembles finis de points adjacents S, caractérisés par une hauteur non nulle et par une connexité, parmi les 8-connexités possibles d'une grille, avec un point de hauteur nulle. A est l'ensemble fini des arêtes reliant deux sommets adjacents appartenant à S. Si G constitue un cycle, alors l'ensemble des points de hauteur nulle encerclés par le cycle appartient à la terre (fig. 30).

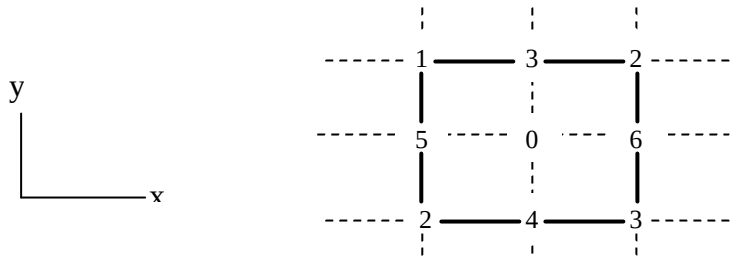


Fig. 30 : Points d'altitude nulle n'appartenant pas à la mer

Dès lors que le littoral est délimité, nous déduisons que l'ensemble des points connexes dont l'altitude est nulle représente la mer. Lors du pré-traitement, nous substituons la valeur de ces points par une valeur spécifique (-¥). En temps réel, la côte et la mer constituent par la suite deux ensembles disjoints dont le rendu est traité individuellement.

Aux limites, pour assurer la continuité au bord du rivage, le bord de côte est fermé par rapport au plan de la mer à l'aide des données bathymétriques (fig. 31). Il en résulte qu'au bord du rivage, les maillages représentant la côte et la mer se superposent. Lors de la construction d'une image, les surfaces cachées sont éliminées grâce à l'algorithme de Z-buffer [Fol99] qui est directement pris en charge par la carte graphique.

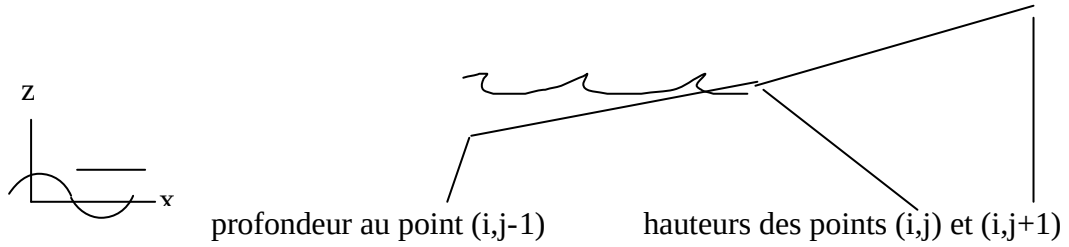


Fig. 31 : Superposition des surfaces de la mer et de la côte

5.4.2 La profondeur et la pente du fond de la mer

Un fichier de relevés bathymétriques est constitué d'une succession de lignes d'iso profondeur triées par ordre de profondeur croissante. Chaque ligne est la suite des points exprimés en coordonnées géodésiques dont la profondeur est identique (fig. 32). Pour exploiter ces données en temps réel, nous effectuons un pré-traitement des fichiers de relevés bathymétriques.

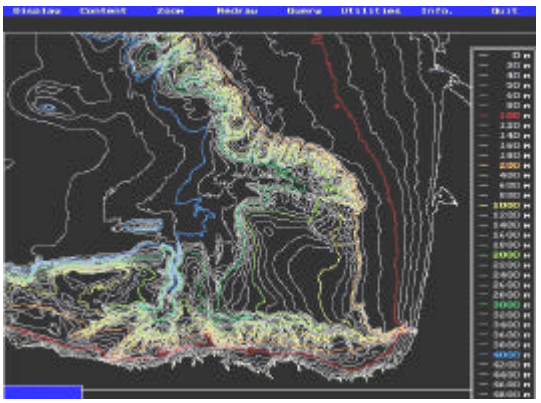


Fig. 32 : Le Golfe de Gascogne

Chaque fichier de relevés bathymétriques est exprimé en une matrice carrée, que nous avons appelé grille de deuxième niveau dans le paragraphe précédent, dont l'élément est la profondeur. Le calcul de la profondeur au point *x* est le calcul d'une moyenne (5.1) :

$$h_x = \frac{(C_1 h_1 + C_2 h_2 + \ldots + C_n h_n)}{(C_1 + C_2 + \ldots + Cn)} \qquad (5.1)$$

avec, pour $1 \leq k \leq n$: $C_k = \frac{1}{\left\| xy_{h_k} \right\|}$

où h_k est la profondeur au point *y* qui est le point appartenant à la ligne bathymétrique *k* le plus proche de *x*, ce qui amène à passer en revue les points consécutifs d'une ligne bathymétrique jusqu'à ce que soit déterminé le bon segment *ii+1* (fig. 33).

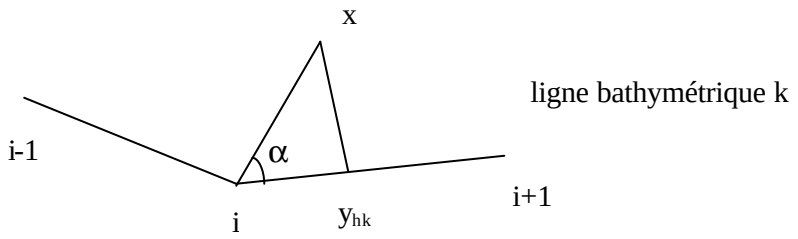


Fig. 33 : Transformation d'une ligne bathymétrique

Les coordonnées du point y_{hk} sont celles du point *i* auquel est ajouté le vecteur iy_{hk} (5.2) :

$$iy_{h_k} = \frac{\left\| iy_{h_k} \right\|}{\left\| ii_{+1} \right\|} ii_{+1} = \frac{\left\| ix \right\| cos(a)}{\left\| ii_{+1} \right\|} ii_{+1} \qquad (5.2)$$

Nous avons également besoin de connaître la pente *a* et la courbe de niveau locale pour appliquer l'algorithme de lancer de vague au point *x*. Dans [TB87], Ts'o et Barsky ont directement utilisé les lignes bathymétriques, telles qu'elles sont représentées sur la figure 32, pour appliquer la loi de Snell&Descartes et étudier les changements de propagation des vagues. Cette approche est peu

satisfaisante car elle manque de précision. Dans [Gon99], J.C. Gonzato a proposé une autre approche que nous avons adoptée. Il échantillonne trois points *A, B, C* situés à une distance arbitraire *dd* de *x* qui constitue un delta-plan (fig. 34). Le delta-plan permet tout d'abord de déterminer la pente a au point *x* ; puis, par interpolation linéaire, il calcule sur les arêtes *AB* et *AC*, les points de profondeur identique à *x*. On obtient, ainsi, la courbe bathymétrique locale.

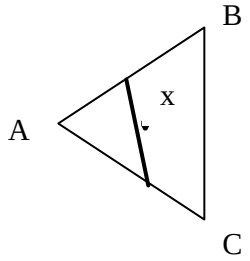


Fig. 34 : Ligne bathymétrique locale

La profondeur *h* et la pente a en un point de l'océan sont utilisées pour construire les *patchs Lancer de vague* qui sont des grilles de troisième niveau. En temps réel, pour contrôler que le navire ne s'échoue pas, nous avons besoin de connaître la profondeur où il se trouve. Pour traiter les extensions que nous avons proposées dans le paragraphe 4.3.2, pour construire un *patch FFT*, nous avons également besoin de connaître *h* et a. Nous constituons alors une grille de deuxième niveau. En temps réel, les données où se trouve le navire sont calculées par interpolation bilinéaire des valeurs des quatre points les plus proches. Nous l'illustrons ci-dessous pour le calcul de la profondeur (fig. 35) :

$$\begin{cases} h_{navire} = h_A + (h_B - h_A)\dfrac{y}{maille_y} \\ h_B = h_3 + (h_4 - h_3)\dfrac{x}{maille_x}, h_A = h_1 + (h_2 - h_1)\dfrac{x}{maille_x} \end{cases} \tag{5.3}$$

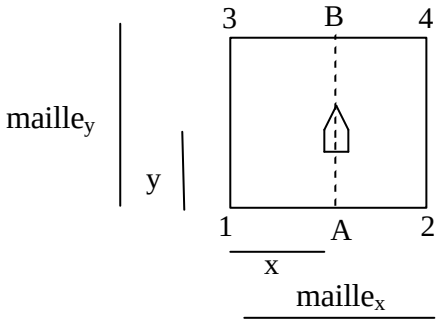


Fig. 35 : Interpolation bilinéaire

5.4.3 Les vecteurs vents et courants de marée

Les données initiales de courants et de vents sont des champs de vecteurs. Pour reproduire des situations exactes, elles peuvent être extraites des logiciels MaxSea développé par la société Informatique & Mer [IM] et ARPEGE développé par le centre de recherche de Météo-France [CNRM].

Concernant ARPEGE, les données de vent sont calculées à une hauteur de 20 mètres au-dessus du sol. La direction et la dimension d'un vecteur sont la direction et la vitesse du vent (fig. 36).

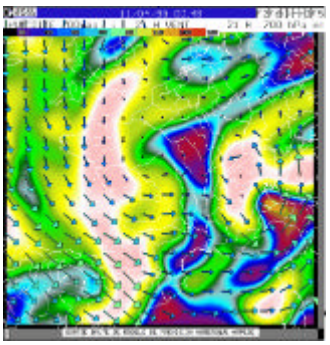


Fig. 36 : Le logiciel ARPEGE

Un premier calcul de moyenne permet d'adapter la résolution des données extraites de MaxSea et d'ARPEGE au format recherché. Puis, pour déterminer la direction et la vitesse du vent, ainsi que celles du courant, au point *x*, nous utilisons à nouveau la méthode d'interpolation bilinéaire comme nous venons de la présenter. Cette fois, les calculs ne portent pas sur un champ scalaire, comme c'était le cas pour la profondeur, mais sur un champ de vecteurs. Nous appliquons, alors, la méthode d'interpolation bilinéaire pour chaque composante du vecteur courant et du vecteur vent.

5.5 Architecture logicielle

Dans notre contexte, la contrainte la plus forte est celle du temps réel (contrainte 1). Elle conditionne le choix de l'architecture logicielle.

Le gain le plus important pour améliorer le taux d'images par seconde est d'exclure les éléments de la scène 3D qui sont situés à l'extérieur du cône de vision. Seulement, comme la caméra est susceptible d'être placée sur la passerelle du navire, ses déplacements sont alors solidaires de celui-ci. En conséquence, les déplacements du navire doivent être déterminés au tout début de la boucle de simulation, préalablement au calcul des six plans constitutifs du cône de vision. L'architecture du logiciel est alors composée de tâches séquentielles facilement identifiables qui s'enchaînent logiquement les unes après les autres, dont la finalité de l'enchaînement est d'exclure les éléments de la scène 3D situés à l'extérieur du cône de vision (fig. 36). Lorsque l'opérateur d'exclusion n'est pas appliqué, il faut attendre plusieurs secondes avant que ne s'affiche la première image.

Concernant la première tâche, le comportement du navire dépend des forces en présence (chapitre 7). Ce sont, d'une part, les forces appliquées directement sur le navire comme la poussée de l'hélice, la dérive du gouvernail, les forces de dérive et de frottement qu'exerce le vent sur le centre de voilure du navire, et les forces qu'exercent les vagues sur la coque du navire. Ce sont, d'autre part, les forces de résistance, de redressement et d'inertie du navire ; on parle alors d'étude de la stabilité du navire. Quant au courant, une simple addition vectorielle de sa vitesse permet d'en tenir compte.

Avec une configuration matérielle comportant plusieurs processeurs, le deuxième moyen pour augmenter le taux d'images par seconde est d'effectuer des calculs parallèles. Les critères pour procéder à la répartition des traitements portent sur la durée d'exécution des processus.

La durée de deux processus devant être synchronisés peut être sensiblement la même. C'est le cas pour les forces exercées par les vagues sur la coque du navire dont le temps de calcul est sensiblement le même que celui nécessaire au calcul de l'ensemble des autres forces que nous venons de détailler. Pour le représenter sur la figure synoptique 37, nous avons mis au même niveau le calcul des forces exercées par les vagues avec celui de l'ensemble des autres forces exercées sur le navire. Les deux processus sont ensuite synchronisés au moment de calculer les variables dynamiques du navire. Le délai de synchronisation constaté est très faible.

Concernant l'allocation et la désallocation en mémoire des données initiales associées aux tuiles traversées ou incluses dans le cône de vision, qui sont effectuées « à la volée » au fur et à mesure que le navire se déplace, ces deux tâches ne sont pas parallélisées comme elles portent simultanément sur la gestion de la même ressource.

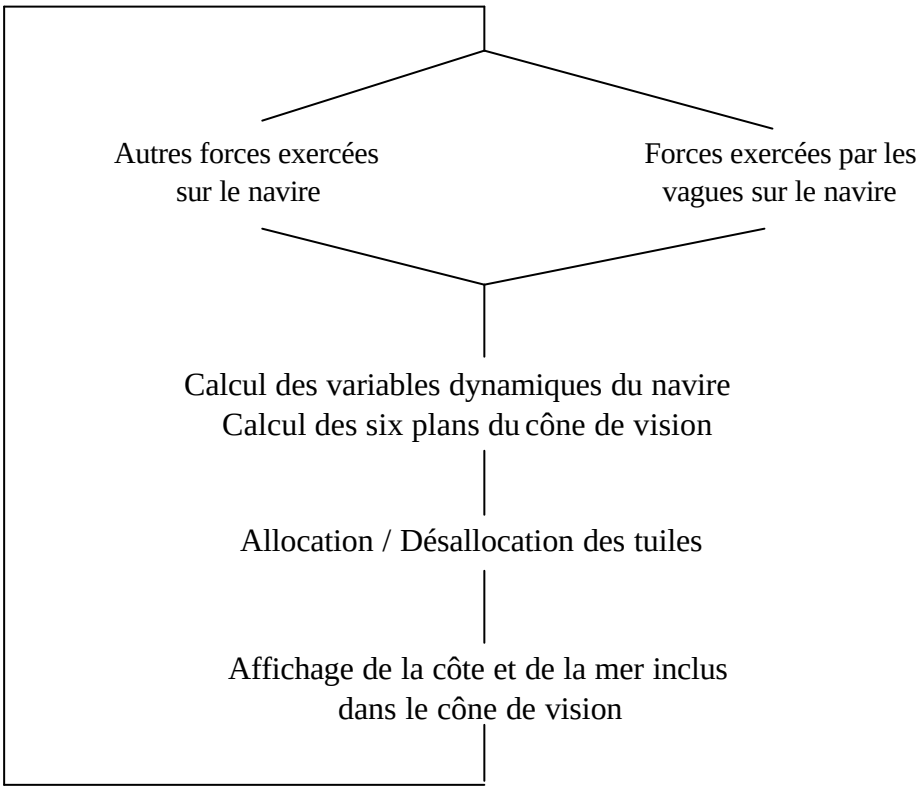


Fig. 37 : Notre architecture logicielle

La durée de la plupart des autres processus étant très inférieure à la durée du processus principal auquel est associé le contexte graphique, il est également possible de les paralléliser. Par exemple, pendant que le processeur principal affiche les maillages de la côte inclus dans le cône de vision, un autre processus est créé pour calculer les métriques d'erreur en vue d'un affichage par

niveau de détails des surfaces de mer. Dans ce cas, il n'existe pas de délai d'attente lors de la synchronisation pour l'affichage des surfaces de la mer incluses dans le cône de vision.

Nous privilégions une programmation multi-thread à une programmation multi-processus, plus rapide comme elle permet d'éviter les temps de recopie des contextes et plus pratique comme la visibilité sur les données est conservée. Chaque fois que nous avons créé un objet «thread», nous avons pu constater des gains mineurs de l'ordre de quelques centièmes de secondes proportionnels à la complexité de la séquence d'instructions associée (forces exercées par les vagues sur le navire, métriques d'erreur, …). Ces gains sont naturellement faibles par rapport à l'opération d'exclusion du cône de vision mais ils ont, néanmoins, un impact sur la fluidité de l'animation.

5.6 Exclusion du cône de vision

5.6.1 Principe général

L'opération d'exclusion du cône de vision est la première fonctionnalité proposée par les outils logiciels que l'on peut qualifier de graphes de scène. Par ordre chronologique de commercialisation, on peut citer les outils développés par la société Silicon Graphics autour du moteur 3D OpenGL comme les outils Open Inventor [OInv] et Performer [Perfor], WorldToolkit [WldTk] développé par la société Sense8 et OpenSG [OpSG] développé par le consortium OpenSG Forum.

Une scène 3D est généralement organisée en une structure arborescente telle que, pour chaque nœud de l'arbre, il devienne possible de construire un volume englobant qui recouvre l'ensemble de ses descendants. Nous invitons le lecteur à se reporter à [Caz97] pour une description détaillée des structures arborescentes les plus utilisées.

En ce qui nous concerne, nous avons choisi une subdivision planaire du paysage en tuiles. Il en résulte de nombreux avantages par rapport à la structure de données des outils de graphes de scène. Dans ce qui suit, nous avons classé ces différents avantages suivant que le test d'exclusion est appliqué au premier niveau de découpage (les tuiles) ou bien au deuxième niveau de découpage (les grilles MNT).

Pour les objets de la scène 3D (navires, maisons, bâtiments, ...) autres que la côte et les surfaces de mer, nous utilisons comme volume englobant le parallélépipède, pour lequel, il suffit de tester si les huit extrémités sont incluses dans le cône de vision pour décider d'afficher l'ensemble des éléments 3D qu'il englobe. Lorsque le parallélépipède englobant est partiellement inclus dans le cône de vision (il existe une arête du parallélépipède dont une extrémité est incluse dans le cône de vision et l'autre extrémité est excluse, ou il existe une arête du parallélépipède dont les deux extrémités sont excluses mais pour laquelle l'arête traverse le cône de vision) les tests d'exclusion se poursuivent de manière récursive (fig. 38).

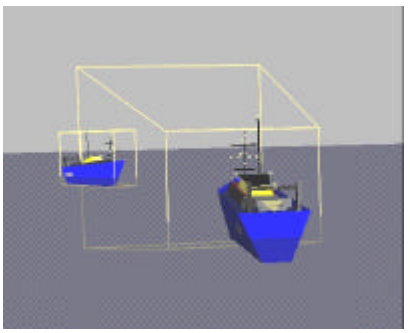


Fig. 38 : Parallélépipède englobant et cône de vision

Pour effectuer les tests d'exclusion, nous utilisons l'équation générale du plan en étudiant deux à deux les six plans du cône de vision avec chacune des deux extrémités d'une arête d'un parallélépipède englobant :

$$Ax + By + Cz + D = 0 \qquad (5.4)$$

Comme un plan coupe l'espace en deux, tout point sera situé au-dessus de ce plan si le signe de l'équation (5.4) est positif ou situé en dessous si le signe est négatif.

5.6.2 Exclusion au niveau des tuiles

Les objets de la scène 3D, qui ne sont pas en mouvement, sont associés à des tuiles. Ils possèdent leur propre mode de subdivision (leur propre structure arborescente).

Tout d'abord, à chaque pas de simulation, nous calculons en premier les tuiles traversées par un ou plusieurs des plans du cône de vision. Nous en déduisons les tuiles incluses dans le cône de vision, comme elles sont délimitées par la première catégorie de tuiles. Il résulte du découpage que chaque ensemble d'objets 3D associé à une tuile incluse est par la suite affiché sans que cela nécessite aucun calcul (avantage 1).

Pour les tuiles traversées par au moins un des six plans du cône de vision, le test d'exclusion porte uniquement sur les objets 3D associés à ces tuiles qui forment un groupe d'objets (avantage 2). Avec un outil de graphes de scène comme l'outil Performer [Perfor], les premiers groupes d'objets sont, par défaut, directement rattachés à la scène 3D elle-même. S'il n'existe pas de niveau d'arborescence entre le premier nœud représentant la scène et ses fils, plusieurs groupes d'objets, alors les tests d'exclusion sont appliqués sur chacun des groupes d'objets.

Il nous reste, cependant, un problème à traiter, œlui des objets de la scène qui sont en mouvement. Cela concerne les navires mais également les animaux marins (dauphins, bancs de poissons qui créent des remous à la surface de la mer, …). Pour ces derniers, nous effectuons un seul test d'exclusion qui consiste à comparer la distance de l'objet avec la portée maximale du cône de vision (plan de coupe «far ») pour ne pas effectuer les tests d'exclusion de leur parallélépipède

englobant. Sur la figure 39, les tests d'exclusion du parallélépipède englobant portent uniquement sur le navire inscrit dans le cercle de portée maximale du cône de vision (avantage 3).

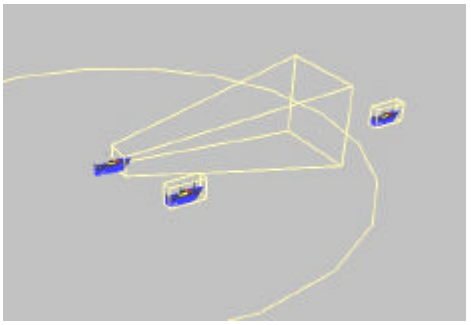


Fig. 39 : Inclusion des navires dans le cône de vision.

5.6.3 Exclusion au niveau des grilles MNT

A chaque pas de simulation, nous parcourons chacune des lignes d'un tableau MNT associée à une tuile traversée par au moins un des plans du cône de vision afin de déterminer les indices des colonnes de début et de fin d'affichage. Les tests d'exclusion portent alors sur les couples de points (i,j) avec $0 \pounds i,j \pounds N$ et $(i+1,j)$ avec $0 \pounds i+1,j \pounds N$ (i pour les lignes et j pour les colonnes) ; si les couples de point sont exclus, on incrémente j. Nous retenons les points exclus les plus proches d'un plan de coupe afin que les fermetures avec les plans du cône de vision soient correctement effectuées (fig. 40).

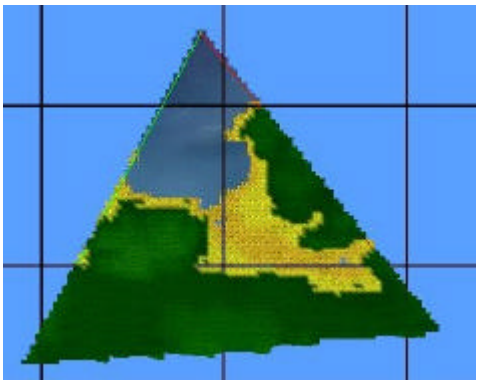


Fig. 40 : Fermeture des surfaces de la mer et de la côte avec les plans du cône de vision.

Pour déterminer les colonnes de début et de fin d'affichage, une autre façon de procéder est de directement calculer l'intersection d'un plan de coupe avec les segments de droite qui relient deux données d'élévation consécutives d'une ligne d'un tableau MNT. Dans ce cas, au lieu d'effectuer des tests d'exclusion, on effectue des calculs d'intersection.

Nous allons voir dans le paragraphe suivant que les colonnes de début et de fin d'affichage auront été prépondérantes quant au choix de la méthode d'affichage par niveaux de détails. Ces deux informations nous permettent, en effet, d'afficher les surfaces de la mer et de la côte comme étant

constituée d'une suite de triangles adjacents. Ce mode d'affichage est extrêmement performant en temps réel. On emploie couramment le terme anglo-saxon de « strip » pour le désigner.

5.7 Affichage par niveaux continus de détails des surfaces de mer

5.7.1 Travaux précédents

Dans [Ögr00], Andreas Ögren propose une classification de toutes les méthodes de rendu de terrains par niveaux continus de détails qu'il répartit en les quatre catégories suivantes : grilles régulières, méthodes de subdivision hiérarchique [Lin96] [Duc97], méthodes par raffinement successif [Hop97] et méthodes traditionnelles comme la triangulation de Delaunay.

Il ne cache pas ensuite ses préférences pour les méthodes de subdivision hiérarchique : la résolution spatiale dépend du relief et de la distance de la caméra. Mais les paysages qui nous sont présentés sont suffisamment variés, i.e montagnes et plaines, pour qu'à chaque image le nombre de polygones à afficher reste à peu près constant quels que soient les déplacements de la caméra. Lorsque le nombre de polygones à afficher reste constant, alors le taux d'images par seconde le reste également.

Il y a des ressemblances entre le rendu de terrains et le rendu des surfaces de mer. Néanmoins, il nous incombe, à chaque pas de simulation, de calculer les «données d'élévation » (hauteurs et déplacements dans le plan de la mer). De plus, nous n'avons pas à traiter des paysages variés : en pleine mer, pour une zone *fetch*, pour un groupe de trains de vagues (une houle), à une qualité de rendu il pourrait être associé une densité presque constante de polygones à afficher pour l'ensemble de la zone de navigation. Dans ces conditions, depuis la passerelle d'un navire, seul l'éloignement de la caméra, qui dépend de l'acuité visuelle de l'homme à la barre, peut nous permettre de diminuer le nombre de polygones à afficher. Au fur et à mesure de l'éloignement, l'homme à la barre distingue de plus en plus mal les vagues capillaires ; au loin, à l'horizon, il ne distingue plus la plupart des vagues de gravité, contrairement au sommet d'une montagne dont le contour inscrit dans le ciel reste distinct.

Les méthodes par raffinement successif et la triangulation de Delaunay présentent l'avantage de ne pas avoir à construire plus de triangles qu'il n'en faut, comme les triangulations portent sur un ensemble de points dont la répartition n'est pas issue d'une subdivision planaire ou d'une subdivision hiérarchique. Cela peut s'avérer judicieux lorsqu'un bord de côte est très escarpé. Néanmoins, construire une surface par le biais de ces méthodes nécessite des calculs en temps réel non négligeables. Aussi, compte-tenu de notre contrainte forte de temps réel (contrainte 1), nous n'approfondissons, dans ce qui suit, que les deux premières catégories de méthodes basées sur l'utilisation de grilles régulières et de structures de données hiérarchiques, pour lesquelles il peut exister des superpositions entre la côte et la mer, mais, pour lesquelles le résultat d'une triangulation est plus rapide grâce au choix préétabli d'un mode de subdivision régulier de l'espace.

5.7.1.1 Subdivision hiérarchique à l'aide d'arbre quaternaire

Peter Lindstrom a été l'un des premiers a présenté une méthode complète de rendu de terrains par affichage de niveaux **continus** de détails [Lin96]. La notion de continuité y est définie de la manière suivante. Il est nécessaire que la géométrie de la surface (l'approximation du terrain) change de façon progressive de manière à ce que l'œil de l'observateur soit trompé en ne se rendant pas compte des modifications apportées. La transition continue entre deux niveaux de détails est plus communément appelée « geomorphing ».

Pour chaque tuile, la structure de données employée est en arbre quaternaire (fig. 41). Chaque nœud de l'arbre représente un carré, décomposable en quatre triangles, à partir duquel une subdivision hiérarchique s'opère récursivement.

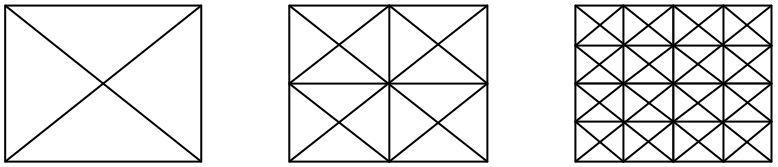


Fig. 41 : Triangulation d'un arbre quaternaire

À l'initialisation, Peter Lindstrom choisit le niveau de subdivision hiérarchique le plus élevé. Au moyen d'une analyse ascendante, il utilise la métrique d'erreur (5.5), plus connue sous le nom de la métrique du delta-segment, pour décider si des opérations de fusion doivent être effectuées.

$$d^2 \frac{\left((c_x - v_x)^2 + (c_y - v_y)^2 \right)}{\left((c_x - v_x)^2 + (c_y - v_y)^2 + (c_z - v_z)^2 \right)^2} \leq \left(\frac{t}{d\,l} \right)^2 \qquad (5.5)$$

où d est le delta-segment, c est la position de la caméra, v est le milieu du delta-segment, t est le seuil de la métrique, d est la distance du point v au plan de projection et l est le nombre de pixels par unité du système de coordonnées 3D projeté sur le système de coordonnées X-Y de l'écran.

La représentation géométrique du delta-segment, qui est projeté sur l'écran, est donnée sur la figure 42. Pour déterminer si les triangles *DABD* et *DBCD* doivent être fusionnés (*DACD* = *DABD Å DBCD*), il est calculé l'erreur commise suivant l'axe z en enlevant le point B. Le delta-segment est alors la valeur absolue de la différence entre la hauteur du point B et le milieu du segment de droite AC : $d_B = ½B_Z - (A_Z + C_Z)/2½$

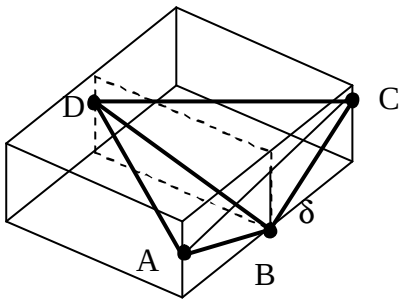


Fig. 42 : Projection du delta-segment

La métrique du delta-segment a été largement commentée depuis.

En temps réel, à chaque pas de simulation, Peter Lindstrom propose une méthode **incrémentale** pour augmenter le niveau de subdivision hiérarchique d'un carré ou fusionner des triangles en un nouveau carré à partir du niveau de subdivision précédent. Toutefois, la méthode incrémentale est difficile à cerner ; son implémentation demande un travail important d'investigations en algorithmique que nous n'avons pas mené.

5.7.1.2 Subdivision hiérarchique à l'aide d'arbre binaire

Dans [Duc97], Mark Duchaineau a également proposé une approche complète d'affichage de niveaux **continus** de détails. Sa méthode paraît plus simple à implémenter que celle de Peter Lindstrom du point de vue algorithmique. On la désigne plus communément sous l'acronyme de ROAM (« Real-time Optimally Adapting Meshes »).

L'approche proposée est aussi une **approche incrémentale** , d'où son efficacité en temps réel. À l'initialisation, pour une tuile visible, un premier maillage est construit ; puis, en temps réel, la géométrie du maillage initial est modifiée progressivement en fonction des déplacements de la caméra. Lors de la simulation, tout saut brusque de la caméra (champ de vision d'une autre personne regardant dans une autre direction) équivaut à une phase de ré-initialisation. Les performances en sont altérées.

La forme géométrique de base est le triangle. Pour subdiviser un triangle, on utilise un arbre binaire, encore appelé « bintree », pour lequel chaque nœud de l'arbre est un triangle dont les fils sont les deux triangles que l'on obtient si l'on divise le triangle père en partant du centre de l'hypoténuse (fig. 43).

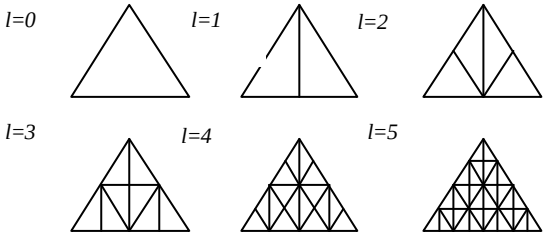


Fig. 43 : Niveaux 0 à 5 d'un « bintree »

Pour décider de manière incrémentale à chaque pas de simulation si l'opération à effectuer est une subdivision, une fusion ou si aucune opération n'est à appliquer, Mark Duchaineau utilise une métrique de base similaire au delta-segment à laquelle il apporte une amélioration. L'erreur commise est transmise récursivement d'un niveau à un autre afin de prendre en considération l'erreur totale commise en enlevant plusieurs points consécutifs.

A chaque nœud de l'arbre, qui correspond à un triangle, est associé un intervalle d'erreur qui tient compte des erreurs commises aux niveaux inférieurs.

La propagation d'une subdivision au sein du maillage afin d'éviter les craquelures est réalisée grâce à la structure en diamant. Un diamant est un losange qui forme quatre triangles adjacents lorsqu'il est découpé par son centre (fig. 44).

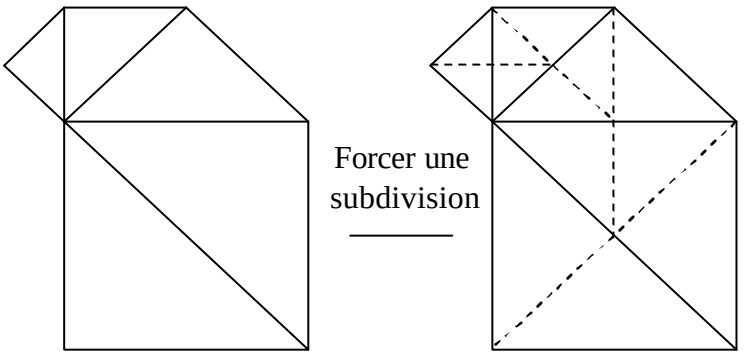


Fig. 44 : Propagation récursive des subdivisons

Enfin, le passage à l'implémentation est également expliqué, ce qui est suffisamment rare pour être souligné. Deux listes de priorité sont utilisées à cet effet. Une pour les triangles à subdiviser et une autre pour les diamants à fusionner, ce qui permet de déterminer le maillage final en un temps optimal.

5.7.1.3 Réseaux irréguliers triangulés en triangles isocèles à angles-droit

Dans [Eva01], Williams Evans commence par expliquer que la méthode d'approximation qu'il propose se situe quelque part entre une sous-grille régulière d'une représentation MNT initiale et un réseaux irrégulier de triangles, basé sur l'algorithme de Delaunay, tel que présenté dans [Rab97].

L'intérêt de la représentation MNT, répondant à notre deuxième contrainte (contrainte 2), est que seule la coordonnée d'élévation est mémorisée. Pour obtenir un niveau de détails, on procède à une subdivision uniforme de l'espace qui se traduit par une grille régulière, pour laquelle il n'est pas nécessaire de mémoriser les cordonnées situées dans le plan du niveau de la mer, puisque le pas de la grille permet de les déduire. Pour les réseaux irréguliers de triangles, la triangulation s'opère en fonction du relief, mais il faut mémoriser pour chaque sommet ses trois coordonnées, cela pour chacune des surfaces incluses dans le cône de vision.

Pour adapter la triangulation à la géométrie du terrain sans avoir à mémoriser les coordonnées des sommets des triangles, Williams Evans propose alors une subdivision hiérarchique d'un carré initial en triangles isocèles rectangles. On obtient ainsi un « bintree » similaire à l'algorithme ROAM décrit dans le paragraphe précédent. L'article ne propose pas une méthode complète d'affichage de niveaux continus de détails mais son intérêt réside dans les explications relatives à la construction de l'arbre binaire. En ce sens, il est très complémentaire de l'algorithme ROAM.

Dans cet article, il est clairement expliqué comment déduire les coordonnées des sommets des triangles en parcourant l'arbre binaire. De la régularité du mode de subdivision hiérarchique, on déduit les coordonnées des sommets des triangles dans le plan situé au niveau de la mer. Simultanément lors du parcours, on construit un index qui permet d'accéder à la coordonnée d'élévation correspondante du fichier MNT.

Les fils de la racine correspondent aux triangles obtenus en divisant le carré initial en deux suivant un axe nord-ouest / sud-est. Les fils suivants sont obtenus en coupant les triangles depuis le milieu de leur hypoténuse jusqu'à l'angle droit opposé (fig. 45).

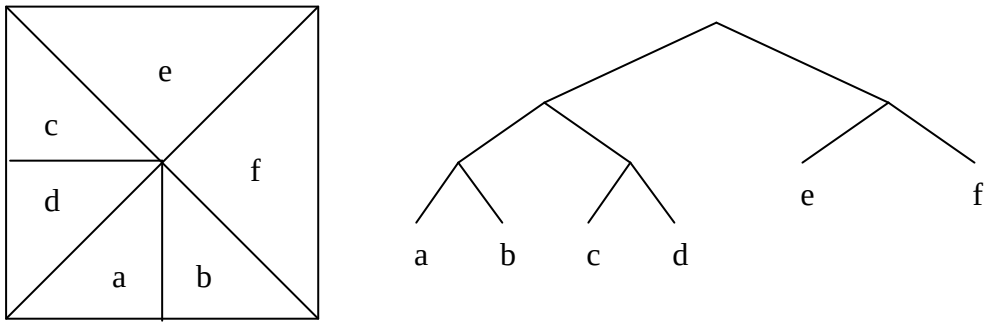


Fig. 45 : « bintree » et coordonnées des sommets des triangles

A titre d'illustration, si (s1, s2, s3) sont les sommets d'un triangle père avec s3 qui est le sommet comportant l'angle droit, les sommets du triangle fils gauche sont (s1, m, s3) et ceux du triangle fils droit sont (s2, s3, m) avec m qui est le milieu de l'hypoténuse du triangle père, c'est-à-dire le milieu du segment s1s2.

Il faut noter que les points sont systématiquement générés dans le même ordre qui est, ici, le sens des aiguilles d'une montre, ce qui permet en phase finale de ne pas afficher les faces cachées des triangles, en activant la fonction de « back face culling » [OGL96]. Cette fonction étudie le signe du résultat du produit scalaire entre le vecteur direction du champ de vision *V* (il forme l'axe *z* du repère définissant l'espace de la vue) et la normale *N* à la face du triangle.

5.7.2 Notre méthode d'affichage de niveaux continus de détails

Après avoir décrit l'état de l'art, nous expliquons dans ce paragraphe la démarche que nous avons suivi pour définir notre méthode d'affichage de niveaux continus de détails. Dans un premier

temps, nous avons mené une **étude comparative** des structures de données de type grille régulière et « bintree ». Pour la construction des arbres binaires, nous avons utilisé l'approche proposée par Williams Evans [Eva01] qui est décrite dans le paragraphe précédent tandis que les grilles régulières sont implémentées par des tableaux à deux dimensions.

Toutes nos expérimentations ont été menées sur un PC bi-processeurs doté de deux processeurs de calcul Pentium III 865 Mhz et d'une carte graphique grand public NVIDIA Geforce 4 Ti 4600.

Nous avons vu que, à chaque pas de simulation, nous commencions par déterminer, lors des tests d'exclusion du cône de vision, quelles étaient pour chaque ligne d'un tableau MNT les colonnes de début et de fin d'affichage. Bien que les caractéristiques du cône de vision soient celles qui sont décrites dans le paragraphe 5.3, nous utilisons une portée maximale (plan de coupe « far ») différente pour l'affichage de la mer de celle du littoral. Pour ne pas écrouler les performances, nous avons choisi une portée maximale de 10 km pour le littoral, et de 5 km seulement pour la mer.

Nous effectuons un premier parcours des lignes des tableaux MNT deux à deux, en partant des colonnes de début jusqu'aux colonnes de fin d'affichage. Lorsque plusieurs points contigus ont une altitude positive, nous constituons une suite de triangles adjacents représentant la côte que nous affichons. Puis, nous effectuons un deuxième parcours des lignes des tableaux MNT deux à deux, en partant toujours des colonnes de début jusqu'aux colonnes de fin d'affichage. Lors du deuxième parcours, il suffit qu'un point, parmi les quatre points contigus délimitant un élément de surface, ait une altitude égale à -¥ (paragraphe 5.4.1) pour ajouter un troisième niveau représentant la mer. Le troisième niveau est le niveau des *patchs FFT* ou des *patchs Lancer de vague*.

Au troisième niveau de découpage, la métrique d'erreur conditionne l'approximation globale d'un *patch*. Dans un premier temps, nous choisissons la métrique la plus basique qui est la distance de la caméra au centre du *patch.* Du résultat de la métrique, nous déduisons le pas d'une grille régulière ou la profondeur d'un arbre binaire. Nous recherchons à afficher au minimum un point tous les mètres près du navire, puis, de moins en moins de points (une résolution spatiale moindre) au fur et à mesure que le centre du *patch* est éloigné de la caméra. Les déplacements des particules d'eau suivant les trois axes sont calculés uniquement pour les ensembles de points que nous venons de constituer.

Lorsque deux grilles adjacences possèdent des résolutions spatiales différentes, des craquelures sont visibles entre elles. Les fermetures entre « bintrees » s'opèrent à l'aide de la structure en diamant. Les fermetures entre sous-grilles s'opèrent à l'aide de la structure «T-vertex », plus simple et moins pénalisante en temps réel que la structure en diamant. Mais, ramené à l'échelle de l'océan, nous avons systématiquement constaté une dégradation des performances lorsque les fermetures étaient effectuées. Les expérimentations qui suivent ont alors été menées sans appliquer l'opérateur de fermeture.

De plus, nous n'affichons pas les faces cachées des triangles en activant la fonction de « back face culling » [OGL96] et nous constituons des suites de triangles adjacents (« strips ») dès qu'il nous

est possible de le faire. Le paradigme qui consiste à lever son crayon lorsque nous dessinons illustre bien le concept de « strip ». Tant que le crayon n'est pas levé, le processeur graphique conserve dans ses registres les coordonnées de deux des sommets du triangle précédent (fig. 46). Lorsque le crayon est levé, la carte graphique procède à des nouveaux transferts sur le bus pour dessiner le triangle suivant. Le temps d'affichage est plus long.

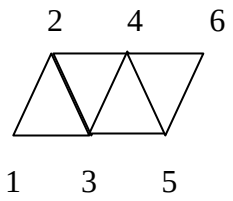


Fig. 46 : Une suite de triangles adjacents

Performances

Les tests de performance (fig. 47) sont effectués en pleine mer afin de ne pas perturber les résultats par l'affichage du littoral. Pour le rendu, il n'est pas appliqué de modèle d'éclairage : pour distinguer correctement la forme géométrique des vagues, la couleur associée à un point du maillage est choisie en fonction de la hauteur du point.

Chaque mesure est exprimée en millionième de seconde. Pour la prélever, nous effectuons la moyenne du temps écoulé durant 100 pas de simulation. Pour recouvrir l'ensemble des surfaces de mer, nous prélevons 8 mesures. La caméra est positionnée sur la passerelle du navire et la vision de l'homme à la barre est dirigée vers la proue du navire. Puis, nous effectuons une rotation de 45 ° sur bâbord jusqu'à revenir à la position initiale.

Points cardinaux	Pas de simulation Grille régulière	Nombre de triangles Grille régulière	Pas de simulation Arbre binaire	Nombre de triangles Arbre binaire
Nord	0.105385	512 576	0.166875	20 548
Nord-ouest	0.103103	495 136	0.188750	24 077
Ouest	0.106250	513 744	0.189231	24 424
Sud-ouest	0.101034	487 736	0.161667	20 017
Sud	0.104815	508 656	0.172727	21 758
Sud-est	0.104615	508 072	0.155789	18 898
Est	0.106216	513 240	0.155556	18 822
Nord-est	0.105000	510 344	0.171538	21 223

Fig. 47 : Etude comparative des structures de données de types « bintree » et grille régulière

Nous pouvons déduire des mesures de la figure 47 que le pas de simulation moyen est de 0.1045522, soit un taux approximatif de 10 images par seconde, avec un nombre moyen de 506 188 triangles pour afficher la mer pour la structure de données de type grille régulière, alors que le pas de simulation moyen est de 0.1702666, soit un taux de 5.8 images par seconde, avec un nombre moyen

de 21 221 triangles pour afficher la mer pour la structure de données de type arbre binaire. Pour ce dernier cas, le taux d'images est moindre mais, surtout, le nombre de polygones utilisés pour représenter la surface de la mer est très faible. Vu d'avion, les différences de densité des deux maillages de la figure 48 illustre bien cet écart entre les deux nombres de polygones. Avec la structure de données de type grille, les vagues sont distinctes même lorsqu'elles sont éloignées de la caméra. Avec la structure de données de type arbre binaire, les vagues sont déformées même à proximité de la caméra. De plus, dans ce dernier cas, si on augmente les seuils de tolérance de la métrique, ce qui revient à augmenter le nombre de polygones, le taux d'images diminue très rapidement.

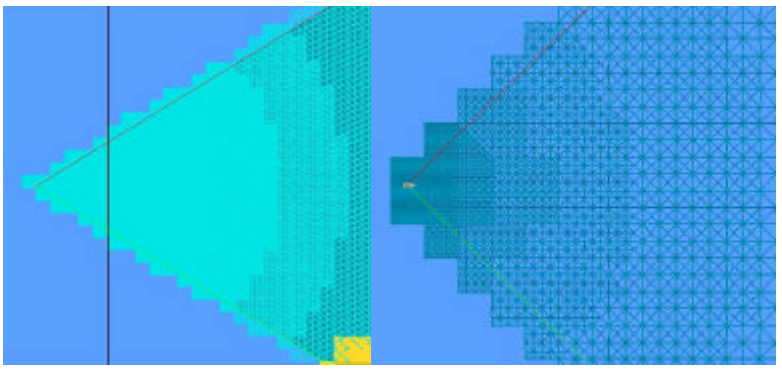


Fig. 48 : Densité des réseaux de triangles

Il y a deux explications à ces écarts de performance. D'une part, le temps de la construction d'un arbre binaire est pénalisant. D'autre part, dans ce cas, il devient rapidement très difficile de construire des « strip » constitués de plus de cinq ou six triangles adjacents.

Dans notre contexte, nous avons donc retenu la structure de données de type grille régulière comme étant la plus adaptée pour représenter fidèlement les vagues océaniques. De plus, chaque ligne d'une grille régulière est affichée comme étant une suite de triangles adjacents plutôt qu'une suite de quadrilatères adjacents. Dans les mêmes conditions d'expérimentation, pour l'affichage des structures de mer incluses dans le cône de vision, on peut constater sur la figure 49 que les performances sont en effet meilleures.

Points cardinaux	Pas de simulation Suite de triangles adjacents	Pas de simulation Suite de quadrilatères adjacents
Nord	0.105385	0.105926
Nord-ouest	0.103103	0.104000
Ouest	0.106250	0.106842
Sud-ouest	0.101034	0.101667
Sud	0.104815	0.105263
Sud-est	0.104615	0.105238
Est	0.106216	0.106429
Nord-est	0.105000	0.105238

Fig. 49 : Etude comparative des suites de triangles et des suites de quadrilatères adjacents

Les craquelures

Au final, nous maintenons ce taux approximatif de 10 images par seconde parce que nous ne fermons définitivement pas les craquelures (fig. 50) entre les grilles adjacentes qui présentent des résolutions spatiales différentes. Pour masquer les craquelures, nous plaçons un plan texturé sous la surface de la mer (fig. 51).

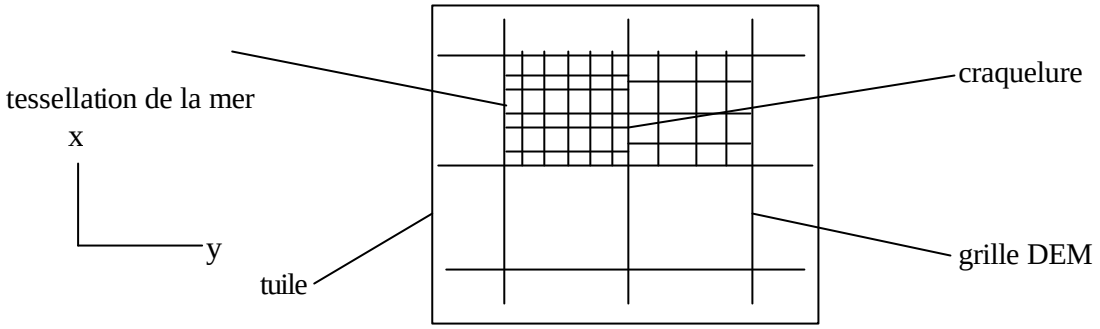


Fig. 50 : Craquelures entre les maillages

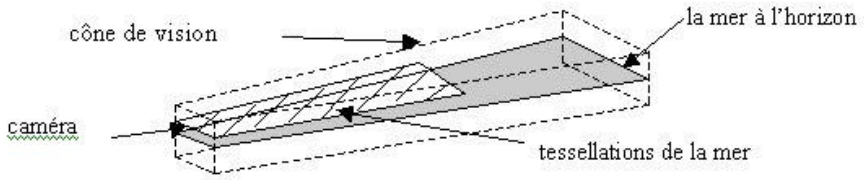


Fig. 51 : Plan texturé sous la surface de la mer

Cette méthode, peu conventionnelle, demande quelques explications. En introduisant un plan texturé sous la surface de la mer, nous tirons ainsi bénéfice de l'algorithme d'élimination des surfaces cachées à l'intérieur du cône de vision, l'algorithme de « Z-buffer » qui pris en charge par l'accélérateur graphique. Lors de la construction d'une image, les pixels dont la couleur serait indéfinie, parce qu'ils correspondent à des craquelures, prennent la couleur d'un texel. Le procédé fonctionne parce que, au moment de l'initialisation, nous positionnons la caméra au-dessus de la mer et nous construisons une première image qui est ensuite utilisée comme texture et parce que les premières craquelures sont très éloignées de la caméra. On évite ainsi d'introduire des fermetures de type « T-vertex ».

De plus, cet artifice présente un autre avantage, comme le plan texturé approxime parfaitement la mer à l'horizon (fig. 51), cela évite d'avoir à masquer la fin du rendu par de la brume comme c'est souvent le cas en synthèse d'images. Nous choisissons comme longueur du plan texturé la portée maximale du cône de vision, qui est ici de 10 km, alors que, lorsque la plus haute vague de gravité n'est plus visible, il n'est plus nécessaire d'afficher les surfaces de mer.

Le réglage des seuils

Le calcul de la distance de la caméra au centre d'une cellule MNT permet uniquement de diminuer le nombre de points à traiter au fur et à mesure que l'on s'éloigne de la caméra. Cette diminution s'opère par paliers successifs (fig. 52).

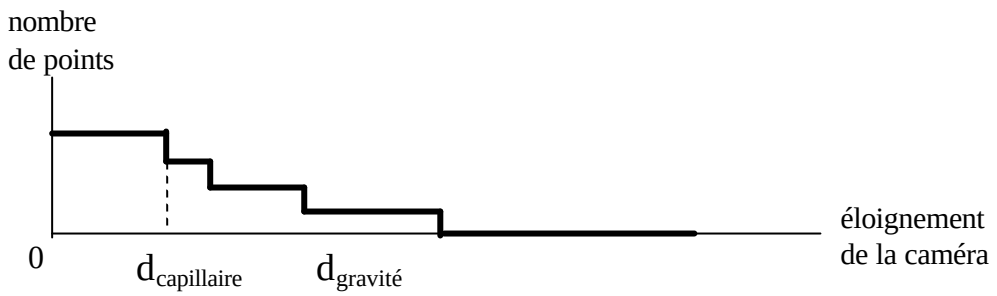


Fig. 52 : Diminution du nombre de points

Les distances $d_{capillaire}$ et $d_{gravité}$ sont les distances maximales respectives au-delà de lesquelles on ne voit plus les vagues capillaires et les vagues de gravité. Si la caméra reste située près du plan de la mer, par exemple à une hauteur correspondante à la vision de l'homme à la barre sur la passerelle d'un navire, leur détermination est un calcul de perspective (5.6) :

$$\begin{cases} \dfrac{h}{d_{capillaire}} = \left(\dfrac{t}{d\,l}\right) \\ \dfrac{H}{d_{gravité}} = \left(\dfrac{t}{d\,l}\right) \end{cases} \qquad (5.6)$$

où h et H sont les hauteurs respectives des vagues capillaires et des vagues de gravité les plus grandes.

Cette approche reste limitée. On ne peut pas, par ce biais, introduire plus de points aux abords d'un rivage lorsqu'une vague déferle. Pour y parvenir, nous avons expérimenté d'autres métriques d'erreur.

Nous n'optons pas pour une approche incrémentale : les déplacements à la surface de la mer sont recalculés à chaque pas de simulation, ce qui n'est pas le cas des données d'élévation en rendu de terrains : nous ne pouvons pas nous baser sur les erreurs constatées au pas de simulation précédent. Nous optons alors pour une **analyse ascendante**. Nous appliquons ainsi un traitement récursif qui porte sur N points, puis sur $N/2$ points, et ainsi de suite jusqu'au test d'arrêt qui est l'utilisation minimale de quatre points. À chaque niveau, nous recherchons l'intervalle maximum d'erreur qui dépend du niveau courant mais également des erreurs commises aux niveaux inférieurs. De cette manière, l'erreur globale, qui est commise en enlevant plusieurs points consécutifs, est propagée d'un niveau à l'autre. Chaque borne de l'intervalle (5.7) dépend des deltas-segment calculés aux niveaux inférieurs et du delta-segment du point courant :

$$\begin{cases} Supd_{xi} = max(Supd_{xi-1}, Supd_{xi+1}, Supd_{xi}) \\ Infd_{xi} = min(Infd_{xi-1}, Infd_{xi+1}, Infd_{xi}) \end{cases} \qquad (5.7)$$

Nous projetons ensuite l'intervalle d'erreur sur l'écran pour connaître l'erreur visuelle commise. Si le seuil de tolérance est dépassé, la récursivité s'arrête à ce niveau ; c'est le deuxième test d'arrêt. La procédure récursive est optimisée car il n'est pas utile de poursuivre les calculs alors qu'un niveau a déjà été atteint ; c'est le troisième test d'arrêt. Nous illustrons le raisonnement sur la figure 53 sur une étude en 1D seulement pour une meilleure compréhension.

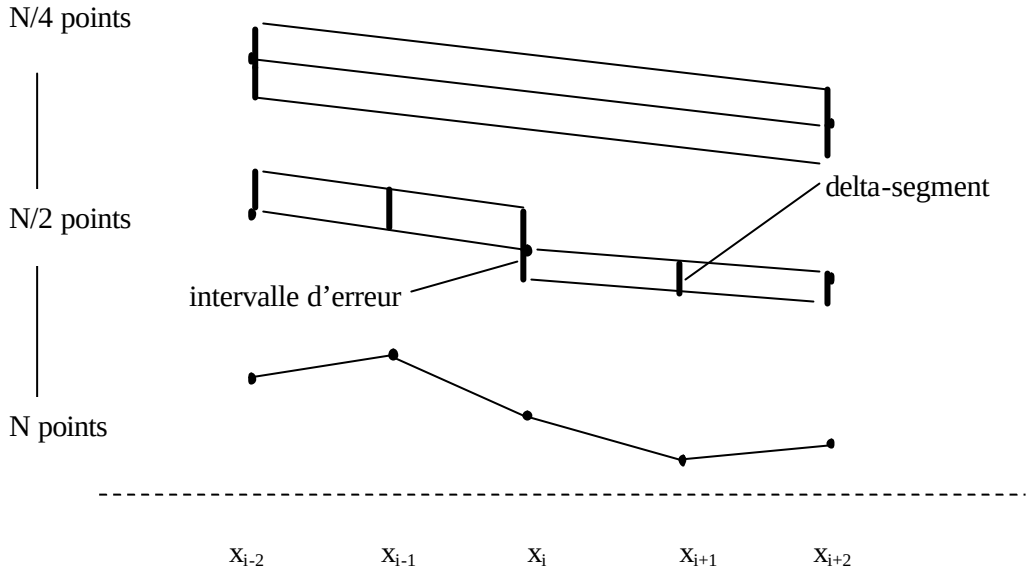


Fig. 53 : Intervalles d'erreur

A partir de la même hypothèse que précédemment, nous effectuons un calcul de perspective pour déterminer la projection de l'intervalle d'erreur (5.8) au point x_i :

$$\frac{\left(Supd_{xi} - Infd_{xi}\right)}{d_{xi}} = \left(\frac{t}{dl}\right) \qquad (5.8)$$

où d_{xi} est la distance du point x_i à la caméra.

5.8 Résumé du chapitre

Dans ce chapitre, nous avons proposé tout d'abord une architecture logicielle adaptée au rendu de scènes océaniques.

Puis, c'est l'utilisation conjointe du découpage du paysage en éléments de surface d'égales dimensions et des parallélépipèdes englobant qui nous a permis d'accélérer les calculs d'exclusion du cône de vision.

Enfin, la méthode d'affichage de niveaux continus de détails que nous avons proposé pour afficher la mer donne de très bons résultats, à la fois sur le plan du rendu et sur le plan des

performances. La mer semble être représentée jusqu'à l'horizon et les formes caractéristiques des vagues de gravité sont très nettement identifiables.

Les performances obtenues avec les grilles régulières sont bien meilleures que celles obtenues avec les arbres binaires. Toutefois, comme nous l'avons déjà précisé, il serait hasardeux de prétendre qu'une méthode est, dans tous les cas, plus performante qu'une autre méthode. Notre approche est intéressante lorsque les dénivellements entre les points varient peu. Une autre application, connexe avec le rendu des surfaces de mer, est le rendu des plages. Par contre, dès que les dénivellements deviennent trop importants, l'approche par grille régulière devient moins adaptée parce qu'elle ne permet pas, au sein d'une même grille, d'afficher plus de triangles à un endroit et moins de triangles à un autre. Il existe un seuil au-delà duquel la rapidité d'affichage ne compense plus un nombre trop élevé de triangles à afficher.

Notre artifice qui consiste à placer un plan texturé sous un maillage pour masquer les craquelures peut être utilisé pour augmenter les performances temps réel dès l'instant où l'envers d'un décor n'est pas visible.

Initialement, nous avions ciblé un rendement de l'ensemble du logiciel et du matériel de 20 images par seconde. Or, en l'état, ce taux d'images ne peut pas être atteint sans diminuer le nombre de polygones à afficher, ce qui aurait comme conséquence de diminuer la qualité du rendu des vagues océaniques. Pour améliorer le rendement de l'ensemble du logiciel et du matériel, nous devons nous doter d'un PC équipé de deux processeurs à haute fréquence, pour les calculs parallèles portant sur le comportement du navire et les métriques d'erreur, et d'une carte graphique « haut de gamme ». Puis, l'étape suivante sera d'exécuter à nouveau des tests de performance.

Chapitre 6

Éclairage temps réel des océans

6.1 Introduction

Nous commençons par présenter les apports successifs portant sur l'éclairage de scènes océanes. Nous souhaitions commencer par constituer un modèle de rendu le plus complet possible.

Par la suite, nos travaux ont porté exclusivement sur l'éclairage temps réel des océans, dont nous rendons compte des résultats obtenus dans ce chapitre. Dans cette optique, il faut tout de suite citer les travaux remarquables réalisés par R. Golias et de R.S. Jensen [Gol01] qui nous ont motivé tout au long de nos recherches sur l'éclairage temps réel des océans. De nombreuses optimisations temps réel sont ainsi présentées dans [Gol01] que nous avons appliqué. Toutefois, nous ne nous sommes pas contentés d'appliquer ces optimisations, nous proposons également une redéfinition du calcul temps réel de la couleur retransmise par la masse d'eau qui, telle qu'il est défini dans [Gol01], est peu satisfaisant.

Comme les fonctions traditionnelles d'éclairage sont fonctionnellement inadaptées à notre cas [OGL96], nous avons rapidement opté pour les facilités offertes par les cartes graphiques programmables. À cet effet, nous avons choisi le langage de programmation Cg [Cg] du constructeur de cartes graphiques NVIDIA [NVI]. Nous présentons ensuite les expérimentations que nous avons pu mener avec le langage de programmation Cg.

6.2 Travaux précédents

. Modèle de Mastin : Dans [Mas87], G.A. Mastin utilise le modèle d'éclairage de Phong [Pho75]. La couleur dépend de l'angle entre la normale à l'océan au point considéré et la direction du soleil dont les rayons sont supposés parallèles. Seulement, les caractéristiques optiques de l'océan sont identiques en tout point de la scène : le coefficient de transmission est fixé à 30% et le coefficient de réflexion à 70%. Le rendu n'est pas réaliste. Le principal défaut de cette méthode, admis par Mastin, est la non-utilisation des coefficients de Fresnel qui permet de déterminer les valeurs de transmission et réflexion en fonction de l'angle d'observation.

. Modèle de Ts'o et Barsky : Dans [TB87], Ts'o et Barsky propose une approximation des coefficients de Fresnel. Pour déterminer la couleur en un point de l'océan, les auteurs appliquent à la surface de la mer deux textures modulées par ce coefficient, textures d'environnement représentant la réfraction et la réflexion. La texture de réfraction est à majorité « bleu-mer » et la texture de réflexion est à majorité « bleu-ciel ».

. Modèle de Fournier Reeves : Dans [FR86], Fournier et Reeves prennent en compte la scène environnante. Le ciel, les nuages et le soleil sont modélisés par une texture d'environnement hémisphérique. Le rendu est obtenu à l'aide du logiciel *Reyes rendering software.* Les auteurs n'ont pas poursuivi leurs efforts pour la modélisation des objets sous l'eau et celle du fond de la mer.

. Modèle de Gonzato : Les travaux de J.C. Gonzato sont plus récents que ceux précédemment cités. Dans [Gon99], l'auteur modélise les objets sous l'eau. Il utilise une représentation spectrale pour traiter l'atténuation de la lumière sous l'eau qui est fonction de la distance parcourue dans le milieu. Son modèle n'est pas exploitable en temps réel. Sur station de travail O2 de Silicon Graphics, le temps de création d'une image varie de 30 minutes à 4 heures en fonction de la taille de l'image.

. Modèle de Tessendorf : Dans [Tes01-1], J. Tessendorf se sert de l'environnement *Blue Moon Rendering Tools* (BMRT) pour effectuer le rendu de l'océan. Dans l'article, la description des phénomènes physiques est très complète alors que le programme *Renderman* présenté est très succinct. Dans ce programme, seulement trois couleurs sont utilisées : une pour le ciel, une pour l'air et une couleur par défaut pour l'eau. Il existe un tel décalage qu'il est difficile de se faire une opinion. Il n'y a d'ailleurs pas plus d'indications sur les performances. Cela peut se comprendre compte-tenu des objectifs commerciaux des travaux. En tout cas, la qualité des images du film « Titanic » est irréprochable.

. **Les travaux de R. Golias et de R.S. Jensen** : Dans [Gol01], R. Golias et R.S. Jensen proposent plusieurs optimisations pour effectuer le rendu de l'océan en temps réel. Ils proposent tout d'abord une approximation simple des coefficients de Fresnel très rapide à calculer en temps réel. Ils utilisent ensuite une texture de type « cube map » pour la réflexion de l'environnement et la réfraction du fond de la mer. Enfin, ils se basent sur la technique du « bump mapping » pour réduire la densité des maillages des surfaces de mer. Par contre, les auteurs font eux-mêmes remarquer qu'ils ne sont pas complètement satisfaits par la couleur «plastique » de l'eau qu'ils obtiennent et par la réflexion spéculaire du soleil à la surface de l'eau.

6.3 Notre modèle d'éclairage temps réel des océans

Le rendu de la mer

Nous commençons par définir notre modèle déclairage. L'intensité I de la lumière, codée par les trois composantes R,G et B, parvenant d'un point x de l'océan à l'œil de l'observateur est donnée par (6.1) :

$$I = (1 - K_{air}(d_x))K_a I_{air} + K_{air}(d_x)\left(I_{soleil} K_{air}(d_{soleil}) K_r \left(k_d (N.S) + k_s (R.S)^n\right) + K_r I_r + K_t I_t \right)$$

où :

. $K_a I_{air}$: la lumière ambiante dans l'atmosphère ; K_a est le coefficient ambiant

. $I_{soleil} k_d (N.S)$: la réflexion diffuse de la lumière ; K_d est le coefficient diffus

. $I_{soleil} k_s (R.S)^n$: la réflexion spéclaire de la lumière ; K_s est le coefficient spéculaire

. K_{air} est le coefficient d'atténuation de la lumière dans l'air qui tient compte de la distance parcourue dans l'atmosphère : d_{air} est la distance parcourue par un rayon de lumière depuis son entrée dans l'atmosphère jusqu'en un point x à la surface de l'océan, d_x est la distance du point x à l'observateur V

. $(K_r I_r + K_t I_t)$ est la couleur en un point x à la surface de l'océan ; I_r et I_t sont respectivement les couleurs de réflexion et de réfraction ; K_r *et* K_t sont respectivement les coefficients de réflexion et de réfraction

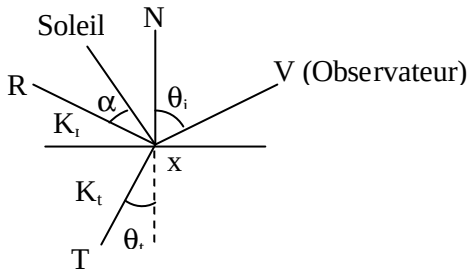


Fig. 54 : Les vecteurs réflexion et réfraction

Pour calculer la réflexion spéculaire de la lumière, nous utilisons l'approximation que Phong a proposée en 1975 [Pho75] : $(cosa)^n$. Nous calculons directement l'angle a comme étant le produit scalaire entre la direction du soleil et le vecteur réflexion R :

$$I_{soleil} k_s (R.S)^n = I_{soleil} k_s (cosa)^n \qquad (6.2)$$

La couleur d'un point de l'océan parvenant à l'œil de l'observateur s'atténue durant sont parcours, ce qui diminue son intensité. Par exemple, vu du ciel, la nébulosité de l'air est plus perceptible que les reflets. L'atténuation $K_{air}(d_x)$ dépend logiquement de la distance du point x, situé à la surface de l'océan, à l'observateur V. Nous l'expliquerons plus en détails dans ce qui suit.

La couleur en un point de l'océan $(K_r I_r + K_t I_t)$ est déterminée à partir de l'algorithme de lancer de rayons, dont le principe général consiste à étudier le trajet inverse de la lumière en partant de l'œil de l'observateur. Tout rayon de lumière incident parvenant à la surface de l'océan sera décomposé en deux : un rayon transmis qui se propage sous l'eau avec un angle de réfraction (q_t) et un rayon réfléchi (R) (fig. 54). L'intensité de chaque rayon (I_r) et (I_t) est diminuée par les coefficients de réflexion (K_r) et de transmission (K_t). Nous parcourons le trajet inverse du rayon réfléchi (R) et de celui transmis par la masse d'eau (T) qui parviennent à l'œil de l'observateur (fig. 54).

Les lois de Fresnel permettent de calculer (K_r) et (K_t). Lorsqu'un rayon lumineux arrive perpendiculairement à la surface, toute son énergie est transmise dans l'océan. À l'opposé, lorsque le rayon est parallèle à la surface de l'océan, toute son énergie est réfléchie. Entre ces deux cas extrêmes, les coefficients de réflexion (K_r) et de transmission (K_t) varient de manière continue. Ils dépendent de la différence d'indice de réfraction entre l'air $(n_b=1)$ et l'eau $(n_a=1.333)$. Ainsi, en mer, les contrastes entre les crêtes et les creux des vagues sont très marqués (fig. 55). Lorsque la direction de la caméra est perpendiculaire à la normale à la surface, le bleu est plus foncé. La couleur dominante est la

couleur retransmise par la couche d'eau. Lorsque la direction de la caméra tend à être parallèle à la normale à la surface, le bleu est plus clair. La couleur dominante est celle du ciel. Il faut également observer sur la figure 55 que les différences de couleur entre les crêtes et les creux, ce qui est du au spectre d'absorption de la lumière dans l'eau.

Fig. 55 : Contrastes entre les crêtes et les creux des vagues

En physique, il existe deux principales approximations des coefficients de Fresnel. Ces approximations donnent de très bons résultats mais elles deviennent rapidement inutilisables en temps réel lorsqu'il s'agit de les appliquer en chacun des points de la surface de l'océan. Nous avons alors finalement adopté la formulation très simplifiée des coefficients de Fresnel proposée dans [Gol01].

$$\begin{cases} K_r = \dfrac{1}{(1+V\cdot N)^8} \\ K_t = 1 - K_r \end{cases} \tag{6.3}$$

Par contre, comme on peut l'observer sur la figure 56, les résultats obtenus à partir de l'équation (6.3) (courbe en pointillé) deviennent moins précis que ceux obtenus classiquement [Gon99] (courbe en trait plein) lorsque le produit scalaire entre la normale à la surface N et l'observateur V tend vers 1. Sur la figure 56, le calcul des coefficients de réflexion est présenté sur l'axe des ordonnées tandis que le produit scalaire entre V et N est présenté sur l'axe des abscisses.

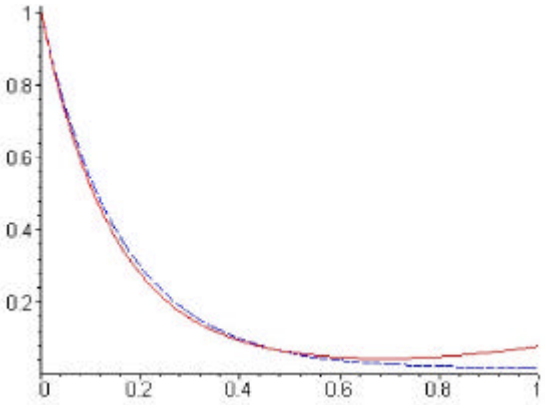


Fig. 56 : Approximation des coefficients de Fresnel

Les équations pour déterminer les directions de réflexion et le rayon de réfraction sont bien connues (6.4). Chacune des démonstrations est détaillée dans [Fol99].

$$\begin{cases} R = 2(V \cdot N)N - V \\ T = N\left(\frac{n_a}{n_b}(V \cdot N) \pm \sqrt{1 - \left(\frac{n_a}{n_b}\right)^2 \left(1 - (V \cdot N)^2\right)}\right) - \frac{n_a}{n_b}V \end{cases} \quad (6.4)$$

où $n_a = 1.333$ et $n_b = 1$.

La visualisation des objets

A partir des directions des rayons *R* et *T*, nous déduisons directement la couleur initiale, exprimée en RGB, des objets de la scène et du fond de la mer. En eaux peu profondes, les marins scrutent le fond de la mer pour ne pas s'échouer. On voit très nettement les reflets des nuages sur la figure 57 (a) et le fond de la mer sur la figure 57 (b). Curieusement sur ces figures, il semble que la mer est volumique et qu'il a été ajouté des effets de transparence, alors que seule la surface de la mer est représentée. Sur cette surface sont reportées les couleurs RGB initiales des objets situés au-dessus et en dessous de la mer.

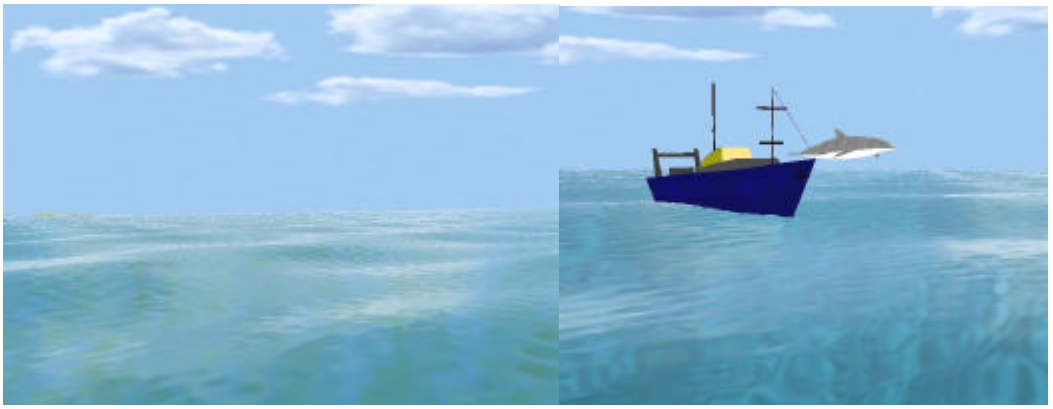


Fig. 57 : Les reflets sur la mer et la visibilité du fond

Le type de texture d'environnement en deux dimensions, plus connu sous le nom de « cube map » [Fer03], est la structure idéale pour retranscrire en temps réel les reflets et le fond de la mer. C'est l'assemblage de six textures en deux dimensions. Chaque texture est la projection d'une vue perspective sur la face d'un cube. L'ensemble des six vues constitue une vue panoramique d'une scène 3D qui entoure un objet ; on parle alors de texture d'environnement. L'avantage du procédé est le calcul immédiat, en temps réel, d'une couleur RGB initiale tout en respectant les effets de perspective. Nous le construisons à l'initialisation à partir des six vues qui constituent la vue panoramique de la scène 3D qui entoure le navire. Au cours de la session de simulation, il peut exister un décalage entre la représentation de la scène 3D et la texture de type « cube map ». Par exemple, de même que [FR86], nous représentons le ciel par une texture hémisphérique où le ciel est « punaisé » sur une moitié de sphère (représenté en filaire sur la figure 58). Lors de la projection sur les faces du cube, cela engendre

une déformation au niveau des arêtes. D'autre part, lorsque le navire se déplace, les faces internes du cube doivent être remises à jour. La fréquence de mise à jour est d'autant plus élevée que le navire est proche des côtes.

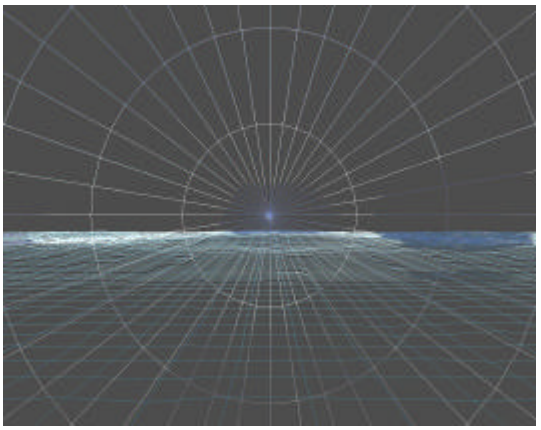


Fig. 58 : Texture hémisphérique du ciel

Un rayon de lumière se propageant dans l'air et sous l'eau subit à la fois les phénomènes de diffusion et d'absorption au contact des particules en suspension. La composée de ces phénomènes est appelée atténuation. Généralement, en infographie, le coefficient d'atténuation K_{air} (ou K_{eau}) est dépendant de la distance parcourue dans le milieu.

Nous définissons I_r, la couleur de réflexion exprimée en mode RGB, comme étant la somme de deux couleurs :

$$I_r = \left(1 - K_{air}\left(d_{objet}\right)\right) I_{air} + K_{air}\left(d_{objet}\right) I_{objet} \qquad (6.5)$$

où K_{air} est le coefficient d'atténuation dans l'air, d_{objet} est la distance de l'objet au point x situé à la surface de l'océan, et I_{objet} est la couleur initiale, exprimée en RGB, de l'objet.

La distance parcourue dans l'air est décomposée en deux : d_{objet} (équation 6.5) et d_x (équation 6.2) correspondant au trajet parcouru par la lumière depuis l'objet jusqu'à la surface de l'océan, puis de la surface de l'océan jusqu'à l'observateur.

Par temps de brouillard, de brume, en présence d'embruns, nous choisissons une atténuation exponentielle d'une intensité RGB dans l'air, alors que, par temps clair, nous choisissons une atténuation linéaire. Cette approche est conventionnelle en synthèse d'images ; elle est généralement utilisée pour introduire du brouillard dans une scène 3D :

$$\begin{cases} K_{air}(d) = 1 - a\dfrac{d}{MaximumVisibilité} & \text{par temps clair} \\ K_{air}(d) = e^{-ad} & \text{autrement} \end{cases}$$

La couleur retransmise par la masse d'eau

Nous définissons I_t, la couleur de réfraction exprimée en mode RGB, comme étant la somme de trois couleurs :

$$I_t = turbidité I_{particule} + (1 - turbidité)\left((1 - K_{eau}(d_{fond}))I_{eau} + K_{eau}(d_{fond})I_{fond}\right) \quad (6.6)$$

où $I_{particle}$ est la couleur exprimée en RGB d'une particule en suspension dans l'eau, I_{eau} est la couleur de l'eau, I_{fond} est la couleur initiale exprimée en RGB du fond de la mer dont l'atténuation K_{eau} dépend logiquement de la distance parcourue dans le milieu (d_{fond}).

Dans l'équation (6.6), nous avons globalement traité la turbidité comme un phénomène additionnel, ce qui est physiquement inexact mais cela nous permet de maintenir un taux constant d'images par seconde. Près d'une rivière, près du rivage, l'eau est chargée de particules en suspension. La couleur dominante que globalement nous ajoutons est celle de la boue près de la rivière, et celle du sable près du rivage.

Le phénomène de diffusion et d'absorption est plus complexe à étudier dans l'eau que dans l'air. Nous ne pouvons pas utiliser directement le modèle de couleur RGB sous l'eau. De ce fait, nous atténuons exponentiellement chaque couleur primaire dans le modèle spectral, puis nous effectuons la traduction en RGB. Si d est la distance parcourue dans l'eau, l'atténuation s'écrit sous la forme :

$$K_{eau}(d) = e^{-a_{R,G,B} d} \quad (6.7)$$

où $a_{R,G,B}$ est un facteur d'atténuation propre à chacune des trois couleurs primaires.

Comme la couleur initiale du fond de la mer I_{fond} est à priori exprimée en RGB, cela nous amène à effectuer deux traductions : une première traduction de la couleur RGB en couleur spectrale qui est atténuée, puis une deuxième traduction de la couleur spectrale en couleur RGB. Le passage du modèle RGB au modèle spectral a été proposé par A.S. Glassner [Gla88]. La transformation d'un modèle spectral vers un modèle RGB a été développé par R. Hall [Hal89].

Abordons maintenant la couleur I_{eau} retransmise par la couche d'eau. Beaucoup de travaux portent sur ce sujet [Pre00]. Selon T. Nishita [Nis94], en l'absence d'objets sous l'eau, l'intensité de la lumière qui parvient en un point P situé sous la mer est la somme de trois couleurs : l'intensité de la lumière ambiante sous l'eau, l'intensité de la lumière qui après réfraction parvient directement depuis la surface au point P, et enfin celle qui parvient indirectement au point P après avoir diffusé au contact d'une particule en suspension dans l'eau.

Tout d'abord, nous avons traité la turbidité comme un phénomène additionnel (équation 6.6). Puis, à des fins de simplification, nous considérons que la couleur provenant d'une direction donnée de la couche d'eau est la somme des intensités qui sont parvenues directement depuis la surface en

tous les points P situés dans cette direction ; au cours du trajet, il y a atténuation de chacune des intensités ; enfin, après diffusion au point P au contact d'une particule en suspension dans l'eau, se produit le trajet inverse pour parvenir jusqu'à l'œil de l'observateur. De ce fait, lorsque la mer est calme, sans vague, nous pré-calculons la couleur sous l'eau dans toutes les directions allant de 0° à 90° ; c'est le calcul d'une intégrale dans une direction donnée. La mer est alors représentée par un plan. Puis, en temps réel, en présence de vagues, la couleur dans une direction quelconque est calculée par interpolation linéaire (fig. 59). Dans [Gol01], il est proposé d'utiliser pour cela l'angle de réfraction du rayon *(T)* (fig. 54) :

$$q_t = arcsin\left(n_a\sqrt{1-(V\cdot N)^2} \right) \tag{6.7}$$

où $n_a=1.333$.

Cette approche nous semble erronée comme q_t est défini par rapport à la normale en un point de la surface de l'océan et non par rapport au plan de la mer. Nous préférons alors utiliser directement l'angle q que forme le rayon *(T)* avec le plan de la mer :

$$q = arcsin\left(\frac{Z_T}{\|T\|} \right) \tag{6.8}$$

où Z_T est la coordonnée du vecteur *(T)* suivant l'axe z.

Fig. 59 : Couleur de l'eau en fonction de l'angle de réfraction

Plus l'angle de réfraction q_t tend vers 90°, vers les profondeurs, plus la couleur bleu est prononcée, alors que vers la surface les nuances de couleur sont plus variées. Lorsque le rayon se propage dans l'eau, la couleur rouge est rapidement atténuée, puis c'est la couleur verte. Seule la couleur bleue s'atténue progressivement. On peut ainsi constater sur la figure 59 que les crêtes des vagues sont plus vertes que les creux.

Les facteurs d'atténuation $a_{R,G,B}$ sont détaillés dans [Jer68].

6.4 La programmation par « shader »

Le processus de construction d'une image est présenté sur la figure 60. Comme chacune des étapes du processus est bien cernée en infographie, nous ne les détaillons pas dans ce paragraphe, mais le lecteur peut se référer à [Fol99] pour plus d'explications.

Un moteur 3D est une machine à états. Si, dans le programme, une primitive graphique d'éclairage est appelée [OGL], alors le processus de construction d'une image est traditionnellement constitué sur la figure 60 des étapes coloriées en bleu pour la géométrie et des étapes coloriées en vert pour le rendu. Seulement, les fonctions traditionnelles d'éclairage sont trop limitatives pour l'éclairage des scènes océanes. Elles ne répondent à aucun des termes de l'équation 6.1. Dans un premier temps, nous avons donc utilisé le processeur de calcul pour implémenter notre modèle d'éclairage.

Avec l'avènement de la quatrième génération de cartes graphiques, des fonctions programmables ont été ajoutées. Il devient ainsi possible de définir un modèle d'éclairage spécifique. Un « vertex shader » est un programme qui est appliqué directement sur la géométrie de la scène 3D (en haut à droite sur la figure 60). Au niveau du tampon image, la couleur finale d'un pixel est alors obtenue par interpolation des couleurs des points de la scène 3D. Mais, il également possible de définir un « pixel shader » (en bas à droite sur la figure 60). Dans ce cas, la couleur finale d'un pixel est également programmée. Nous avons choisi le language Cg (*annexe E*) du constructeur NVIDIA [NVID] pour écrire nos programmes de rendu.

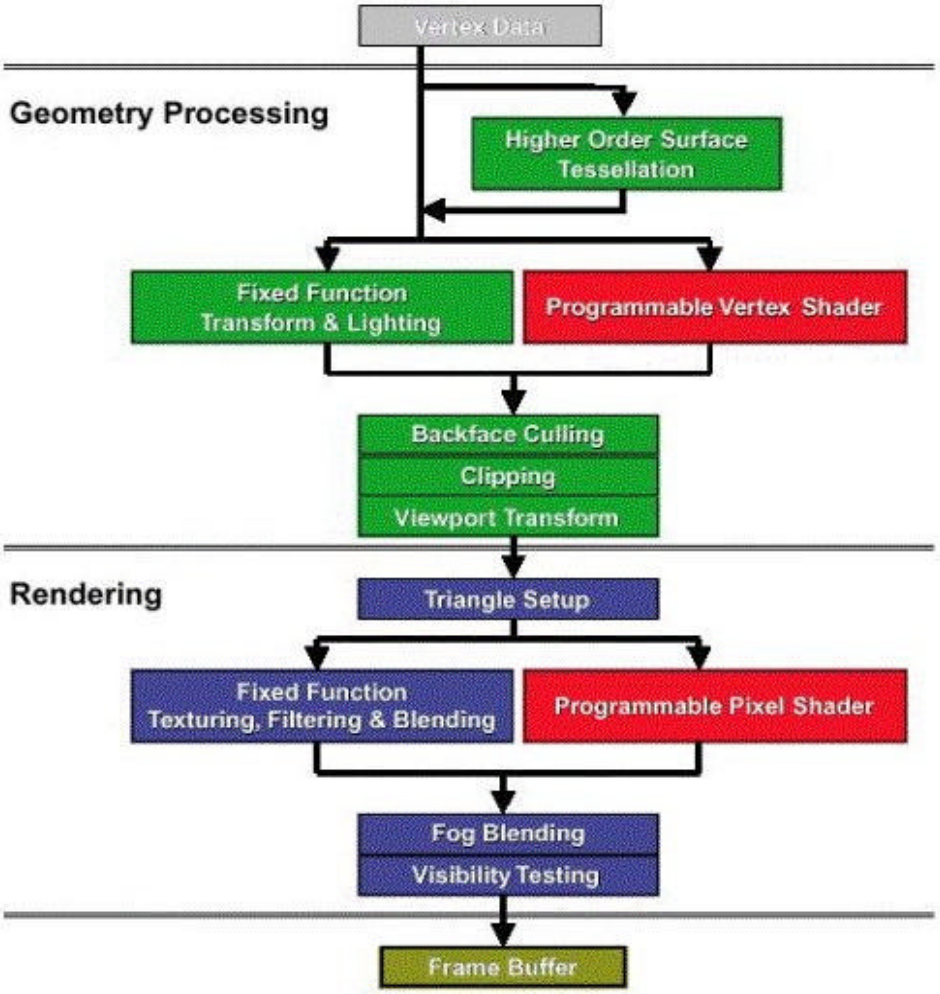


Fig. 60 : La programmation par « shader » [ATI]

Performances

Nous avons renouvelé l'expérimentation décrite dans le paragraphe 5.7.2 mais cette fois en comparant les performances obtenues lorsque l'éclairage de la scène océane est supportée par la carte graphique (GPU) avec celles obtenues lorsque l'éclairage est supportée par le processeur de calcul (CPU).

Comme pour l'expérimentation précédente, on positionne le navire en pleine mer afin de ne pas perturber les résultats par l'affichage du littoral. Le cône de vision est celui qui est défini dans le paragraphe 5.3 avec une portée maximale de 10 km pour l'affichage du littoral et une portée maximale de 5 km pour l'affichage de la mer. De la même manière que précédemment, chaque mesure est exprimée en millionième de seconde. Pour la prélever, nous effectuons la moyenne du temps écoulé durant 100 pas de simulation. Pour recouvrir l'ensemble des surfaces de mer, nous prélevons 8 mesures. La caméra est positionnée sur la passerelle du navire et la vision de l'homme à la barre est dirigée vers la proue du navire. Puis, nous effectuons une rotation de 45 ° sur bâbord jusqu'à revenir à la position initiale.

Points cardinaux	Rendu GPU	Rendu CPU	Nombre de triangles
Nord	0.106364	0.235849	512 576
Nord-ouest	0.103478	0.228333	495 136
Ouest	0.106833	0.236251	513 744
Sud-ouest	0.101429	0.225714	487 736
Sud	0.105172	0.234348	508 656
Sud-est	0.105000	0.234138	508 072
Est	0.106667	0.236000	513 240
Nord-est	0.105758	0.235625	510 344

Fig. 61 : Etude comparative du rendu effectué par le CPU ou la GPU

Nous pouvons déduire des mesures de la figure 61 que le pas de simulation moyen est de 0.105087, soit un taux un peu inférieur à 10 images par seconde, avec un nombre moyen de 506 188 triangles lorsque l'éclairage de la mer est prise en charge par la carte graphique, alors que le pas de simulation moyen est de 0.2332821, soit un taux un peu supérieur à 4 images par seconde, pour le même nombre moyen de triangles lorsque l'éclairage de la mer est prise en charge par le processeur de calcul. Certes, il fallait s'attendre à ce que les résultats soient meilleurs lorsque la carte est graphique est sollicitée mais il faut souligner que l'écart est important. D'autre part, si on compare les mesures avec celles de la figure 47, prélevées dans les mêmes conditions mais pour lesquelles l'éclairage de la scène n'était pas effectuée, on s'aperçoit que les résultats sont presque invariants. Nous pouvons ainsi programmer des « shaders » sans altérer les performances.

Le langage Cg est un dialecte du langage C. Il est doté d'une bibliothèque mathématique très complète et d'une bibliothèque de textures, ce qui permet de réaliser des prototypes très rapidement ;

un simple appel à une primitive permet de construire une texture de type « cube map » ou de type « bump map » et le code écrit est très synthétique. Par contre, seules les dernières cartes graphiques, comportant un grand pouvoir d'expression, peuvent tirer partie de ce langage de programmation de « haut niveau » (critique 1). Pour les séries de cartes graphiques qui précédent la série GeForce FX, le nombre d'instructions autorisées dans un « pixel shader » est très limité. Comme notre carte graphique est une GeForce 4 Ti 4600, cette limitation nous a fortement pénalisés tout au long de nos expérimentations que nous avons du d'ailleurs mener individuellement. À ce propos, un seul exemple de « vertex shader » et de « pixel shader », parmi tous les programmes que nous avons écrits, est présenté en *annexe E*.

Le flot de données de l'équation (6.1) est décrit sur la figure 62.

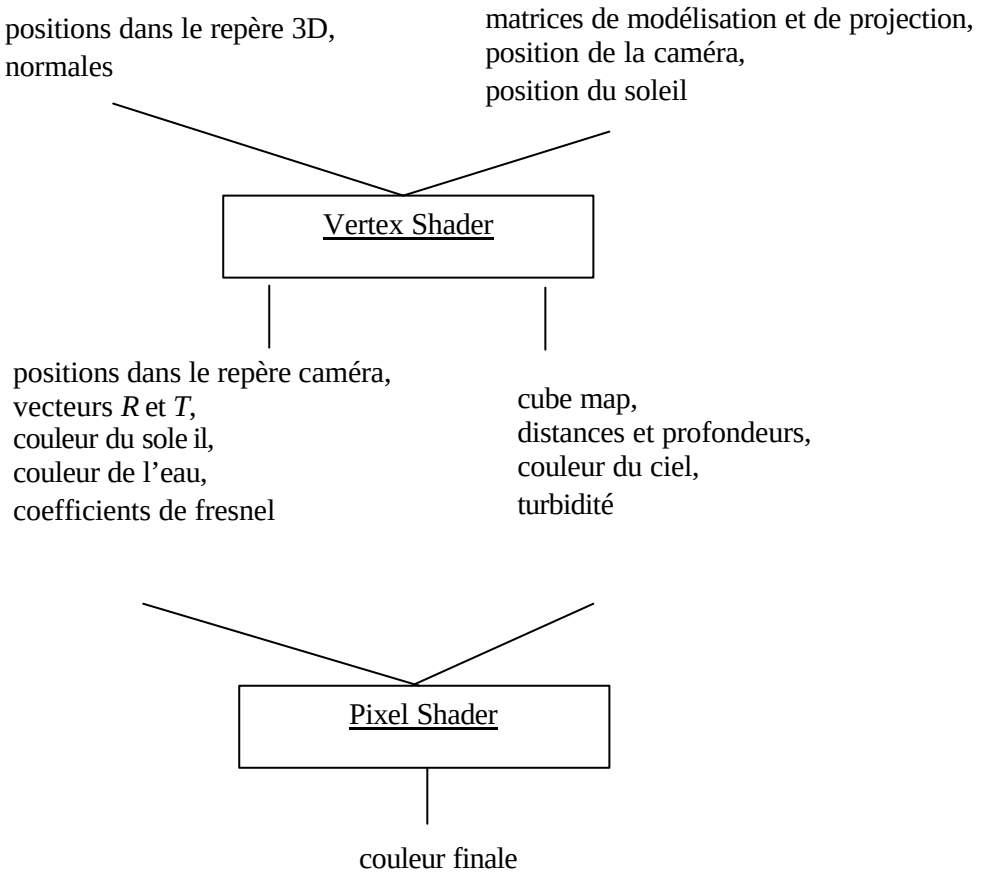


Fig. 62 : Paramètres des programmes d'éclairage

La qualité obtenue est pourtant meilleure lorsque la couleur finale d'un pixel est programmée dans un « pixel shader ». On peut le constater sur la figure 63. Lorsque la réflexion spéculaire du soleil est calculée en un point du maillage, la couleur finale d'un pixel est déterminée par interpolation bilinéaire (fig. 63 a). Lorsque le produit scalaire entre la direction du soleil et le vecteur *(R)* est calculé pour un pixel donné, le rendu est beaucoup plus précis (fig. 63 b). Chaque pixel possède sa propre couleur, distincte d'un pixel contigu. Là encore, quelle que soit la solution choisie, les performances ne varient pas.

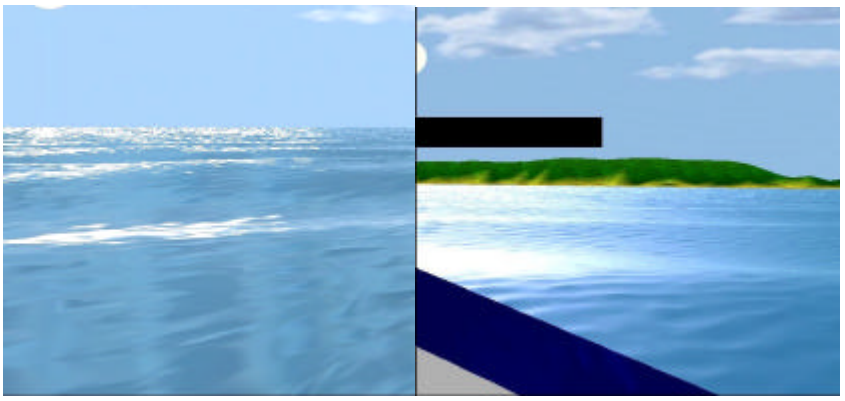


Fig. 63 : La réflexion du soleil dans un « vertex shader » ou dans un « pixel shader »

L'autre critique porte sur l'absence de documentation (critique 2). Dans ces conditions, il est parfois délicat de cerner le service proposé par une fonction bibliothèque. Heureusement, un guide de programmation vient de paraître [Fer03].

6.5 Les textures et la technique du bump-mapping

Dans la version du simulateur installé au lycée maritime de Ciboure (*annexe A)*, les remous à la surface de l'eau sont introduits par le biais de simples textures 2D dès lors qu'un banc de poissons est proche de la surface de l'eau.

Concernant le prototype, nous avons vu que pour traiter les vagues de gravité (basses fréquences) nous avions choisi une résolution spatiale d'un point tous les mètres (chapitre 5). Cette résolution ne nous permet pas de traiter les vagues de courtes longueurs d'onde (de hautes fréquences). Pour les introduire, nous choisissons la technique de «bump-mapping » que Blinn avait proposé en 1968, et qui est décrite dans [Fol99] (p. 744). L'idée, très à propos dans notre contexte, est d'améliorer la densité des détails apparents sans, pour autant, alourdir les maillages des surfaces de mer. Le moyen d'y parvenir est de modifier le calcul des normales des pixels.

En nous appuyant sur la programmation «multi-thread », nous créons un *patch* de quatrième niveau à l'aide d'une texture de normales. À chaque pas de simulation, nous utilisons une unique inverse FFT pour calculer la superposition des mouvements de vagues de courtes longueurs d'ondes ; le champ d'amplitudes est déterminé à partir du spectre théorique que nous avons proposé dans le paragraphe 4.4. La position initiale est une position quelconque inscrite dans le plan d'une mer parfaitement calme ; la position exacte est précisée ultérieurement.

On substitue ensuite les données de hauteurs par les données de normales. La texture de 256x256 texels, équivalente à une densité apparente d'un point tous les 3.9 mm, est ensuite transmise en argument au « pixel shader ».

L'algorithme que nous souhaitons ensuite appliquer, pour le traitement simultané des vagues de Basses Fréquences (BF) et des vagues de Hautes Fréquences (HF), est le suivant :

Pour la scène 3D

TransformeRepereVue(L) ; Normalise (L) ;
TransformeRepereVue(V) ; Normalise (V) ;
// on calcule la normale en un point du maillage
N = CalculeNormalesBF() ;
glNormal (N) ;
// on construit le *patch FFTHF* de quatrième niveau inscrit dans le plan de la mer
PatchFFTHF = ConstruitTextureNormalesHF() ;
// on precise maintenant les coordonnees des vagues de courtes L
// la texture de normales est alors « punaisee » sur une suite de strip
// (autant de repetitions de la texture que d'éléments adjacents)
glTexCoord (s, t) ;

Pour chaque vertex shader

TransformeRepereVue(N) ;

Pour chaque pixel shader

CalculeRepereVue (N, BF_u, BF_v) ;
// Extraction des normales des vagues de courtes L
Extraction(HF_u, HF_v) = TextureNormales (s, t) ;
CalculeRepereVue (HF_u, HF_v) ;
// approximation de Blinn
N' = N + HF_u (N x BF_u) + HF_v (BF_v x N) ; N' = Normalise(N') ;
CalculeEquation6.1(N', V, L) ;

Fig. 64 : Algorithme de « bump-mapping »

où *(N x BF_u)* et *(BF_v x N)* sont des produits vectoriels, *BF_u, BF_v, HF_u et HF_v* , sont les vecteurs tangentes à la surface de la mer (fig. 65).

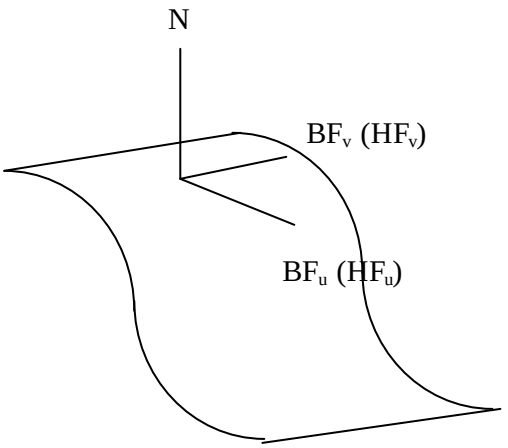


Fig. 65 : Les vecteurs tangentes à la surface de la mer

Nous n'avons pas pu aller au bout du raisonnement à cause de la limite très restrictive du nombre d'instructions autorisées dans un « pixel shader » qui nous était imposé.

6.6 L'écume

Pour un rendu plus réaliste, la dernière étape consiste à ajouter l'écume. L'approche qui nous semble la plus réaliste est celle qui est proposée dans [Tes01-01] (fig. 66).

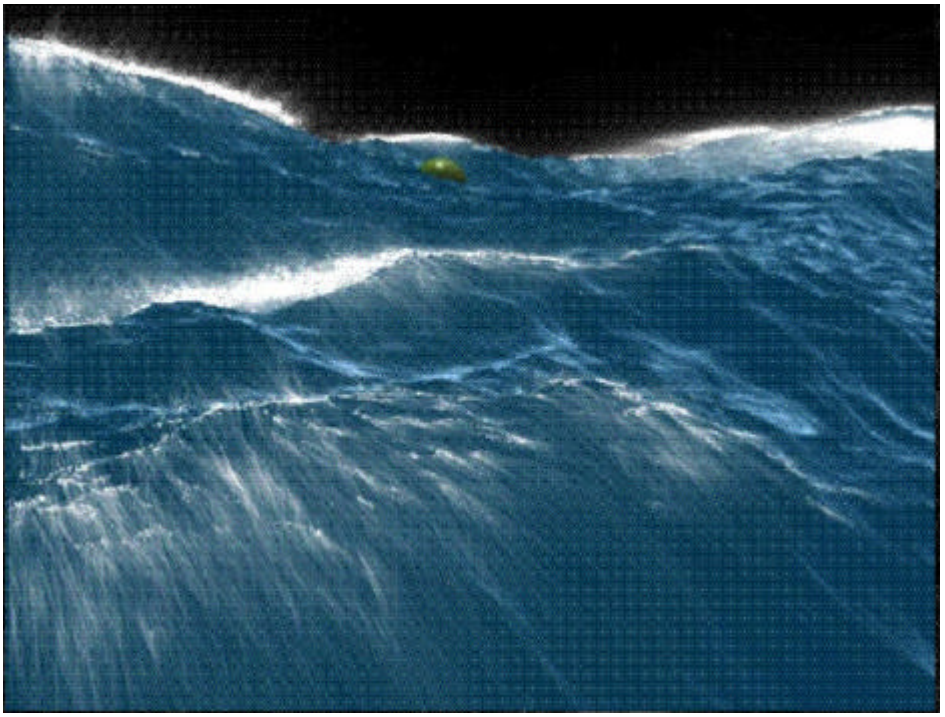


Fig. 66 : L'ajout de l'écume [Tes01-2]

Pour simuler la tempête, pour changer la forme des vagues, J. Tessendorf a proposé d'ajouter le paramètre global l lui permettant de contrôler les déplacements dans le plan $D(x,t)$ au point x. À chaque temps de cycle, il calcule le déterminant de la matrice Jacobienne suivante :

$$J(x) = \begin{bmatrix} J_{xx}(x) & J_{xz}(x) \\ J_{zx}(x) & J_{zz}(x) \end{bmatrix} \qquad (3.9)$$

où $J_{xx}(x) = 1 + l\,\frac{\partial D_x(x)}{\partial x}$, $J_{xz}(x) = l\,\frac{\partial D_x(x)}{\partial z}$, $J_{zx}(x) = l\,\frac{\partial D_z(x)}{\partial x}$, $J_{zz}(x) = 1 + l\,\frac{\partial D_z(x)}{\partial z}$

Lorsque le déterminant *det(J)* devient négatif, il se produit une boucle au point *x* ; on en déduit que la valeur de l est trop élevée.

Nous n'avons pas retenu cette approche. Mais, cependant, calculer le déterminant de la matrice Jacobienne (sans le paramètre l) s'avère utile pour ajouter de l'écume (lorsque le *det(J)* ® *0*).

C'est une indication pour construire une texture 2D avec une densité de blanc à déterminer (en mode « DECAL » [OGL] avec $Couleur_{out} = Couleur_{In}*(1-Alpha_{tex}) + Couleur_{tex}*Alpha_{tex}$ où $Couleur_{tex}$ est blanche et où $Couleur_{In}$ est obtenue à partir de l'équation 6.1, où $Alpha_{tex}$ est égal à 0 ou 1).

Dans [Pre00], S. Premoze, bien qu'il n'ait pas présenté de résultats, a proposé d'utiliser la formule de Monahan [Mon86] pour calculer la densité de blanc que doit comporter la texture :

$$f = 1.59 10^{-19} \|W\|^{2.55} e^{0.0861 (T_{eau} - T_{air})} \qquad (3.10)$$

où $||W||$ est la vitesse du vent exprimée en m/s, T_{eau} et T_{air} sont la température de l'eau et celle de l'air exprimées en degrés Celsius.

À aucun moment, y compris dans des conditions d'ouragan (force 12 sur l'échelle de Beaufort), nous n'avons constaté de boucles à la surface de la mer en utilisant le modèle (3.36). Le cas se produit lorsque la profondeur diminue dans le modèle (4.16). Pour y remédier, nous testons systématiquement si le grand axe d'une ellipse ne dépasse pas la limite supérieure fixée par Gerstner (paragraphe 3.3.2.2). Dès que sa valeur est bornée pour chaque vague, le cas ne se reproduit plus.

6.7 Résumé du chapitre

Dans ce chapitre, nous avons proposé un modèle complet d'éclairage de scènes océaniques que nous avons implémenté à l'aide du langage Cg [Cg] du fabricant de cartes graphiques NVIDIA. De cette manière, l'éclairage n'est pas à la charge du processeur de calcul mais elle est directement traitée par la carte graphique. Nous avons pu constater que, par ce biais, nous n'altérions pas les performances présentées dans le chapitre 5.

Les travaux proposés par [Gol01] portant sur l'éclairage temps réel des scènes océaniques se sont avéraient très pertinents. Nous pensons à l'approximation des coefficients de Fresnel ou encore à l'utilisation de la texture de type «cube map» pour la réflexion des objets de la scène 3D et la réfraction du fond de la mer. Nous avons proposé une amélioration du calcul temps réel de la couleur retransmise par la masse d'eau. Contrairement à [Gol01], la couleur en un point de la surface de la mer que nous obtenons n'a pas une allure plastique (fig. 59) et la réflexion spéculaire du soleil à la surface de la mer paraît réaliste (fig. 63 à droite).

Le nombre très restreint d'instructions autorisées dans un «pixel shader» par notre carte graphique nous a fortement pénalisés. Au cours de nos expérimentations, nous avons pu constater que la qualité du rendu était optimale lorsque la couleur définitive d'un pixel était calculée au niveau « pixel shader », au dernier stade du processus de construction de l'image. De la sorte, chaque pixel possède sa propre couleur, qui ne résulte pas de l'interpolation des couleurs des points du maillage. Les images de mer obtenues sont très réalistes. Mais, au moment de l'intégration de l'ensemble des couleurs au niveau le plus fin, nous avons rencontré un problème insoluble à résoudre du au nombre très restreint d'instructions autorisées.

Nous sommes, aujourd'hui, dans l'attente de l'acquisition d'une carte graphique NVIDIA GeForce FX 5900 pour réaliser l'intégration finale mais, surtout, pour achever nos travaux portant sur le «bump mapping» afin de traiter en temps réel l'ensemble des ondes dues au vent dont les périodes s'étendent jusqu'à environ 30 secondes (paragraphe 4.4).

Chapitre 7

Effet des vagues sur le comportement du navire

7.1 Introduction

Nous n'apportons pas de contributions directes aux travaux présentés dans ce chapitre. Il nous semblait, néanmoins, important d'aborder le problème de tenue à la mer d'un navire parce qu'une modélisation géométrique seule des effets des vagues sur le navire conduit à un rendu physiquement non réaliste de son comportement. Si les forces de résistance peuvent être approximées, ce n'est pas le cas pour les forces directement exercées sur la coque du navire par les vagues. De plus, l'inconvénient majeur d'une modélisation géométrique est qu'elle ne tient pas compte des forces d'inertie du navire.

Il existe principalement deux grandes approches aujourd'hui référencées par la communauté scientifique en mécanique du navire et beaucoup de méthodes s'y rapportant. Nous décrivons, ici, une seule de ces méthodes parce qu'elle donne des résultats acceptables en temps réel et parce qu'elle s'appuie sur la forme linéaire des équations de Navier-Stokes. Elle est plus connue sous le nom de méthode par tranches « strip theory » ou sous l'acronyme STF du nom de ses inventeurs.

7.2 Le calcul de la pression sur la surface mouillée de la coque

Le comportement d'un navire dépend des forces en présence. Ce sont, d'une part, les forces appliquées directement sur le navire comme la poussée de l'hélice, la dérive du gouvernail, les forces de dérive et de frottement qu'exerce le vent sur le centre de voilure du navire, et les forces qu'exercent les vagues sur la coque du navire. Ce sont, d'autre part, les forces de résistance, de redressement et d'inertie du navire ; on parle alors d'étude de la stabilité du navire. Quant au courant, une simple addition vectorielle de sa vitesse permet d'en tenir compte.

Concernant les forces exercées directement par les vagues sur la coque du navire et les forces de résistance hydrodynamique, on parle plus généralement du problème instationnaire de tenue à la mer et du problème stationnaire de résistance de vagues. Ces deux problèmes intéressent fortement les hydrodynamiciens depuis plusieurs dizaines d'années.

La résistance de vagues, plus simple, fut la première à être traitée dès 1898 par J.H. Michell [Mic98], puis par Havelock en 1913 [Hav13]. Il s'agit de déterminer les efforts stationnaires dus au fluide, l'eau, sur une carène avançant à vitesse constante sur un plan d'eau calme. Les efforts globaux se décomposent en une force de frottement d'origine visqueuse (environ 30 % des efforts globaux) et des efforts de résistance de vagues proprement dit. La résistance de vagues représente la perte d'énergie engendrée par la création du champ de vagues autour du corps.

Le problème de tenue à la mer consiste quant à lui à déterminer les efforts instationnaires agissant sur une carène avançant avec une vitesse moyenne et à en déduire, sous l'effet des vagues, ses

mouvements autour de sa position moyenne. Deux grandes approches sont généralement utilisées pour résoudre ce problème : l'une dite fréquentielle, l'autre dite temporelle. L'approche fréquentielle consiste à déterminer la réponse du corps pour chaque fréquence du spectre de vagues, puis à reconstituer les efforts en additionnant les contributions (les surfaces de mer résultent d'une simulation spectrale qui n'est valable que sous les hypothèses admises au chapitre 3 : vagues de faibles amplitudes, spectre d'amplitudes, superpositon des mouvements, ...). L'approche temporelle, qui nécessite beaucoup plus de temps de calcul, s'effectue à partir de l'état de repos du navire en étudiant progressivement sa réponse impulsionnelle. Le lecteur trouvera dans [Bec00] un tour d'horizon complet des quinze méthodes existantes pour résoudre le problème de tenue à la mer du navire, qui se rattachent à l'une ou l'autre des deux grandes approches que nous venons de citer.

Les forces de résistance et de tenue à la mer s'étudient au niveau de chacune des parties élémentaires de la surface mouillée de la coque. La somme des forces est le calcul de la force hydrodynamique Hk, suivant les directions $k=2,3,4,5,6$ (fig. 67) correspondants respectivement à la dérive latérale, l'élévation, le gîte (roulis), le tangage et le lacet (giration) ; la force suivant la direction $k = 1$ est considérée comme étant négligeable. Elle est déterminée en intégrant la pression sur la surface mouillée de la coque (6.1). L'approche fréquentielle nous permet de simplifier son écriture en utilisant l'équation de Bernouilli (3.7) ; on obtient :

$$H_k = -\iint_S p n_k \, d_S = -r \iint_S \left(\frac{\partial f}{\partial t} + \frac{1}{2} |\nabla f|^2 + gz \right) n_k \, d_S \qquad (6.1)$$

pour laquelle, la normale n_k est orientée positivement dans la direction du fluide, S est la surface mouillée de la coque et f est le potentiel total du fluide

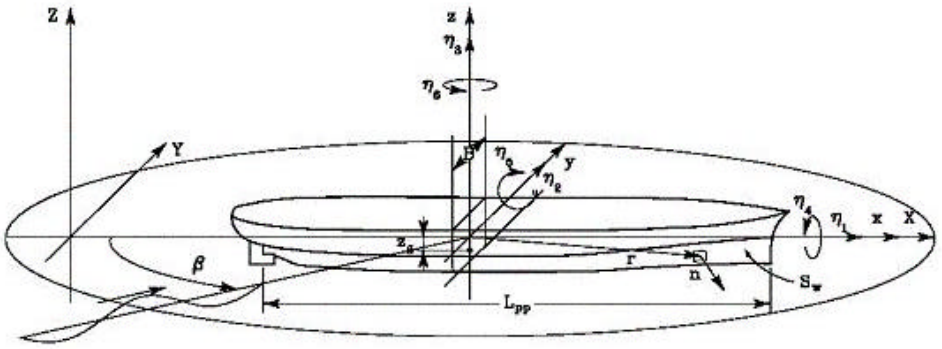


Fig. 67 : Les degrés de liberté du navire [Ped00]

Le potentiel total du fluide f se décompose en un potentiel de vitesse stationnaire indépendant du temps f_S (résistance de vagues) et un potentiel de vitesse dépendant du temps f_T (tenue à la mer) :

$$f(x,y,z,t) = -Ux + f_S(x,y,z) + f_T(x,y,z,t) \qquad (6.2)$$

où U est le vecteur courant.

Le potentiel de vitesse dépendant du temps f_T se décompose à nouveau en le potentiel incident des vagues f_I et en d'autres potentiels, que nous ne décrivons pas ici, mais qui résultent du contact avec la surface mouillée de la coque (diffraction, ...). Enfin, il n'existe pas un seul potentiel incident mais autant de potentiels incidents que de vagues de gravité en présence (approche fréquentielle).

La méthode des tranches permet de représenter le problème tridimensionnel par une somme de problèmes bidimensionnels. Elle a été introduite par Kroukovsky en 1955 [Kro55] pour calculer la tenue à la mer d'un navire sans vitesse. Le traitement d'un navire qui avance a été ensuite proposé en 1969 par Ogilvie et Tuck [OT69], et amélioré en 1970 par Salvesen [STF70]. On parle maintenant plus communément de la méthode STF. Elle est aujourd'hui utilisée en simulation d'entraînement maritime et permet de prédire en temps réel les mouvements d'un navire [Bec00].

Il existe beauoup d'implémentations de la méthode des tranches. Une implémentation très complète est proposée par Tommy Pedersen [Ped00]. Dans [Ped00], l'architecture logicielle est présentée, les algorithmes et le code source sont commentés, et, en annexe, on peut trouver un manuel utilisateur. Les résultats obtenus sont classés par catégorie de navires (cargo, ...). Par manque de temps, nous n'avons pas pu pousser plus loin nos investigations.

Par la suite, il serait intéressant de recréer les situations qui peuvent amener à ce que le navire chavire [Cie02-2]. Par exemple, comme on peut l'observer sur la figure 68, jusqu'au point d'inflexion situé aux environs de 40°, le moment de redressement transversal d'un navire est d'autant plus fort que le gîte est important. Le navire se redresse rapidement. Par contre, au-delà du point d'inflexion, le moment de redressement compense de moins en moins. Le chavirement est alors d'autant plus rapide que le gîte est fort. Les facteurs de risque s'aggravent en fonction de la répartition de la charge à bord et lorsque le navire est en pêche.

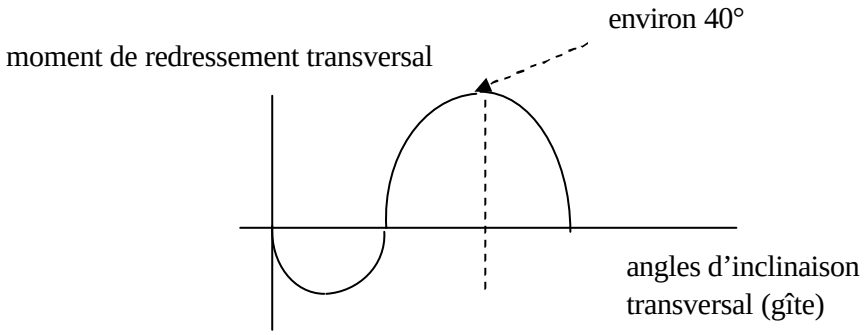


Fig. 68 : Courbe de stabilité transversale d'un navire

Conclusion

Dans ce document, concernant la modélisation des vagues océaniques, nous avons tout d'abord proposé un modèle unique et complet pour simuler en temps réel la forme et la propagation d'une onde de surface (**la houle**) depuis la pleine mer jusqu'au rivage qui prenne en compte les effets du courant de marée et de la profondeur. Les trajectoires des particules tournent sur un cercle en pleine mer, puis sur une ellipse près des côtes dont le grand axe s'oriente progressivement parallèlement à la pente du fond de la mer. La forme et la propagation du train de vagues en un point sont pré-calculées à partir d'un algorithme que nous avons appelé «algorithme de lancer de vague à partir d'un arc de cercle » ; l'algorithme est applicable sur une large étendue et le rendu est géographiquement réaliste (position initiale de la zone de *fetch*, point géodésique courant défini par la hauteur de la marée et par une mesure bathymétrique, ...). Au cours des pré-calculs, les données en un point sont mémorisées au sein de grilles 2D qui recouvrent l'ensemble de la zone de navigation. En temps réel, seul le paramètre temps varie, l'onde de surface est reconstituée en 3D.

Nous avons ensuite proposé une extension du modèle de J. Tessendorf, basé sur la FFT inverse, pour simuler en temps réel la génération des vagues de gravité sous l'effet du vent (**la mer du vent)** qui prend en compte ponctuellement les effets de la profondeur et du courant de marée. Les effets les plus visuels sont le changement de la forme des vagues due aux modifications des trajectoires des particules d'eau et la variation du champ des hauteurs du au courant ou à la profondeur. L'extension proposée reproduit plus fidèlement les *états de mer* en fonction des conditions de navigation (conditions de vent sur l'échelle de Beaufort (*annexe B*), courant de marée, profondeur) que le modèle initial.

Nous avons enfin proposé un spectre théorique pour les vagues de courtes longueurs d'onde qui peut être utilisé en temps réel conjointement avec les modèles de houle ou de mer du vent.

Concernant la simulation temps réel, nous avons tout d'abord proposé une architecture logicielle adaptée au rendu de scènes océaniques. Puis, pour exploiter nos modèles de vagues océaniques en temps réel, nous avons proposé plusieurs optimisations.

Concernant les calculs parallèles au sein d'un même contexte graphique, il est vrai que, aujourd'hui, nous ne retirons pas tous les bénéfices d'une configuration matérielle constituée de deux processeurs. Néanmoins, si nous avions à faire un choix matériel, nous maintiendrions cette configuration. Nous avons vu que l'architecture du logiciel est composée de tâches séquentielles facilement identifiables qui s'enchaînent logiquement les unes après les autres, et dont la finalité de l'enchaînement est, avant tout, d'exclure les éléments de la scène 3D situés à l'extérieur du cône de vision. Cependant, une deuxième analyse montre que chaque tâche séquentielle peut être décomposée en deux tâches parallèles. C'est le cas pour le calcul des forces s'exerçant sur le navire ; c'est encore le cas pour le calcul des *patchs* des vagues de gravité (basses fréquences) et ceux des vagues de courtes

longueurs d'onde (hautes fréquences) ; c'est toujours le cas pour le calcul des métriques d'erreur (côte et mer).

C'est donc l'exclusion des éléments de la scène 3D situés à l'extérieur du cône de vision qui procure le gain le plus important. L'utilisation conjointe d'une subdivision planaire du paysage en « tuiles » et des parallélépipèdes englobant nous a permis d'accélérer les calculs d'exclusion des éléments 3D autres que les surfaces de mer et les surfaces de la côte. Pour ces derniers éléments, un simple calcul, au deuxième niveau de découpage (les fichiers MNT), permet de déterminer les colonnes de début et de fin d'affichage.

Enfin, notre méthode d'affichage de niveaux continus de détails des surfaces de mer donne de bons résultats, à la fois sur le plan du rendu des vagues océaniques et sur le plan des performances. La mer semble être représentée jusqu'à l'horizon et les formes caractéristiques des vagues de gravité sont très nettement identifiables. Dans notre contexte, les grilles régulières sont plus performantes que les méthodes de subdivision hiérarchique. Grâce au calcul des intervalles d'erreur en un point, l'observateur ne se rend pas compte des modifications apportées. La métrique d'erreur est pertinente tant que la caméra reste située à proximité du plan de la mer. Elle n'a plus de sens aux limites lorsque la vue est une vue aérienne.

Notre artifice qui consiste à placer un plan texturé sous un maillage pour masquer les craquelures est réutilisable dès lors que l'envers d'un décor n'est pas visible.

Concernant l'éclairage temps réel des océans, nous avons tout d'abord proposé un modèle complet d'éclairage de scènes océaniques que nous avons implémenté à l'aide du langage Cg [Cg]. De cette manière, l'éclairage n'est pas à la charge du processeur de calcul mais elle est directement traitée par la carte graphique. Nous avons pu constater que, par ce biais, nous n'altérions pas les performances malgré une qualité grandemment améliorée. L'absence de défauts est manifeste lorsque la couleur définitive d'un pixel est calculée au niveau « pixel shader », au dernier stade du processus de construction de l'image ; chaque pixel a alors sa propre couleur, qui ne résulte pas de l'interpolation de couleurs des points du maillage ; au lieu d'uniformiser les couleurs, leur diversité est conservée. Encore très surprenant est le fait que, lorsque nous choisissons une résolution spatiale plus fine des surfaces de mer, leur éclairage n'altère pas les performances, ce qui laisse entrevoir la potentialité de ce mode de programmation pour les applications comme les simulateurs d'entraînement ou les jeux vidéos.

Nous avons ensuite poursuivi les travaux de nos prédécesseurs sur l'éclairage temps réel des océans en proposant une amélioration du calcul temps réel de la couleur retransmise par la masse d'eau. Le reflet sur l'eau sur la scène environnante et la réfraction du fond sont instantanés grâce à l'utilisation d'une texture de type « cube map ». La couleur en un point de la surface de la mer que nous obtenons n'a pas une allure plastique et la réflexion spéculaire du soleil à la surface de la mer paraît réaliste. Dans les années à venir, nous espérons que le qualificatif de « terne » ne sera plus utilisé pour désigner le rendu des images de mer des simulateurs d'entraînement maritime. Nous

sommes, aujourd'hui, dans l'attente de l'acquisition d'une carte graphique NVIDIA GeForce FX 5900 pour terminer les travaux d'éclairage.

L'ensemble des travaux, qui ont été présentés dans ce mémoire, est rassemblé au sein d'un prototype écrit en langage de programmation C++ et en OpenGL [OGL96], disponible sur PC sous système d'exploitation Linux. L'étape suivante est l'industrialisation du prototype, qui nécessite principalement de réécrire les instruments de navigation et de pêche de gl en OpenGL [OGL96]. Viendra enfin, ensuite, le moment du retour d'expérience des stagiaires en formation.

Fig. 69 : La salle de classe : le poste enseignant et les passerelles

Le simulateur d'entraînement à la navigation et à la pêche est un outil pédagogique, homologué par l'Inspection Générale de l'Enseignement Maritime (IGEM), conçu pour que les élèves des lycées maritimes assimilent au mieux les connaissances théoriques et pratiques liées à l'exercice des métiers de la mer.

Imaginé par des enseignants, l'outil permet de transposer les situations professionnelles de la réalité, tout en s'affranchissant du coût et de la difficulté de la mise en œuvre dans des conditions normales, dans une salle de classe à partir d'une solution purement informatique basée sur la reconstitution d'images de synthèse et sur une solution réseau.

L'enseignant y définit les exercices de simulation conformément aux référentiels de formation en vigueur et y assure la conduite de l'apprentissage grâce à un poste dédié qui lui permet de conserver à tout moment la maîtrise du système. Tandis que les élèves se retrouvent à bord de navires devant une passerelle de simulation et sont mis dans des situations, qui correspondent à leurs aspirations professionnelles, pour lesquelles le système leur confère un degré d'interactivité total. Ils naviguent et pêchent, en temps réel, dans une région qui leur est familière mais dont les conditions météo varient, avec les navires des autres élèves ainsi que les navires introduits par l'enseignant.

Le poste enseignant supporte cinq fonctionnalités qui sont la préparation des exercices de simulation et des questionnaires à choix multiples qui y sont associés, la gestion de la base de données des élèves, l'affectation des élèves à un poste de simulation et la supervision d'une session de simulation.

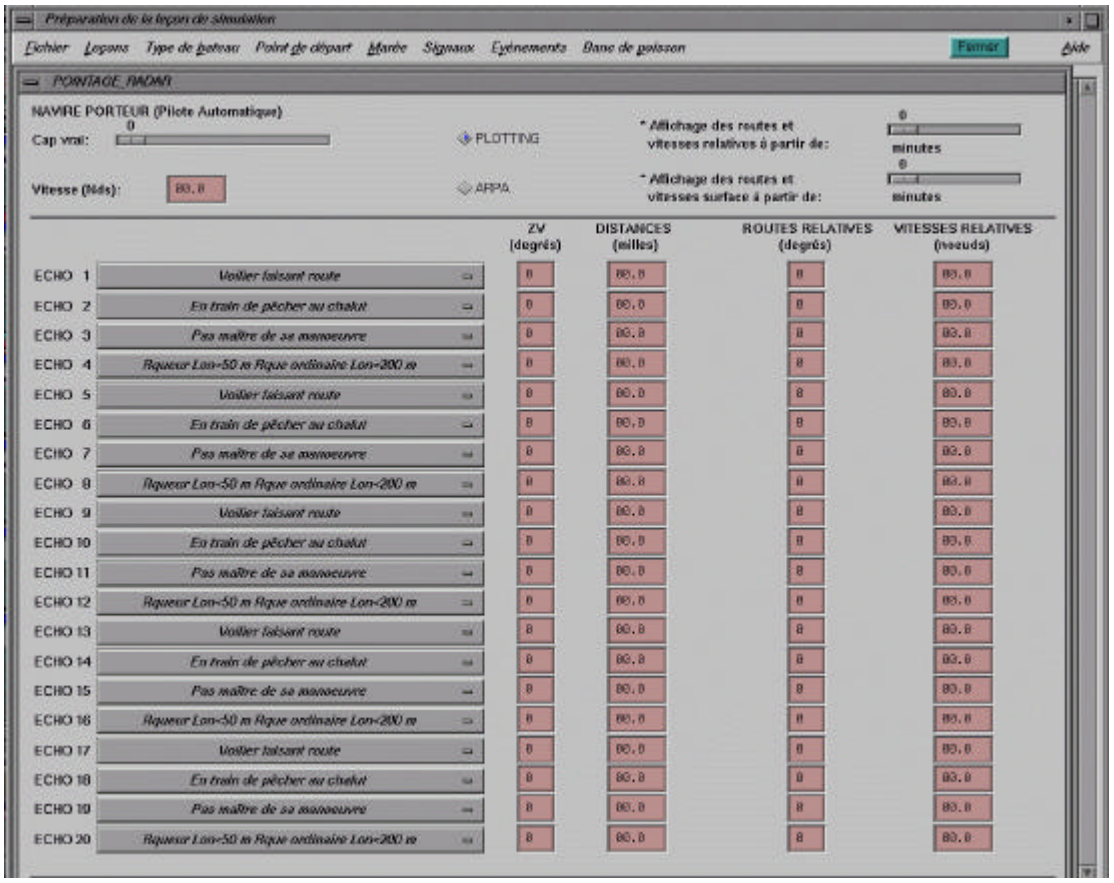


Fig. 70 : Préparation d'un exercice de pointage radar

Le simulateur est tout d'abord pour un enseignant un support idéal pour formaliser et retranscrire des connaissances. Cette formalisation est réalisée chaque fois qu'un exercice de simulation est défini. Par souci de professionnalisation de l'enseignement général, les exercices introduits sont classés dans un premier temps par matière enseignée, puis, par souci pédagogique, dans un deuxième temps par niveau de complexité croissante. Aussi, trouve-t-on des catégories d'exercice se rapportant à l'apprentissage de la signalisation maritime, à l'exécution de manœuvres, à la pratique de la navigation, à l'utilisation d'un radar, à la pratique de techniques de pêche industrielle et à d'autres catégories encore. Aussi, trouve-t-on des niveaux de complexité croissante liés à la concentration de navires, à l'état de la mer et à la force du courant, à la force du vent et aux conditions de visibilité (pluie, brouillard, …).

Le navire à piloter peut être en navigation un cargo, un pétrolier ou encore un remorqueur et en pêche un chalutier ou un senneur tandis que l'enseignant peut introduire jusqu'à 50 navires supplémentaires au mouillage et jusqu'à 20 navires supplémentaires en mouvement.

Le choix du point de départ de la leçon dépend de la zone géographique dans laquelle se trouve le lycée maritime (Golfe de Gascogne, Méditerranée, …).

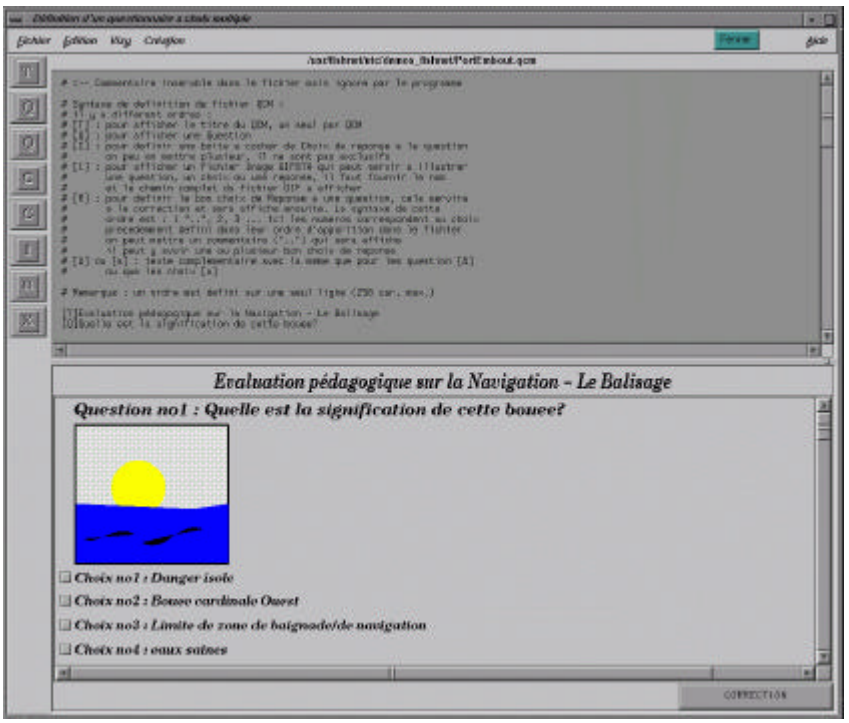


Fig. 71 : Préparation d'un questionaire à choix multiple

A chaque exercice de simulation est associé un Questionnaire à Choix Multiples afin de s'assurer préalablement de l'état des connaissances générales d'un élève.

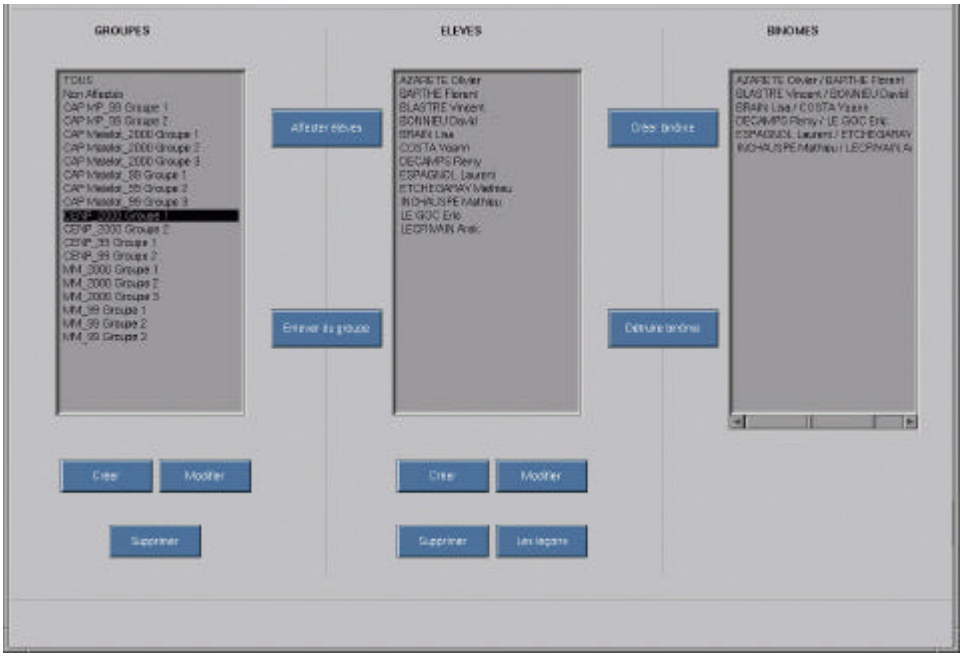


Fig. 72 : Gestion de la base de données des élèves

L'administration des élèves est réalisée au moyen d'une base de données. Cette base de données contient tous les noms des élèves du lycée. Une première répartition par groupe y est effectuée en fonction des diplômes préparés; puis, au sein de chaque groupe, une deuxième répartition

par binôme y est également effectuée en fonction du poste de travail sollicité à bord d'un navire. La base de données permet de faciliter le suivi de l'apprentissage de l'élève et propose à cet effet plusieurs fonctions comme l'affichage de l'historique des leçons qu'il a effectuées.

Au début d'une session de simulation, un binôme d'élèves appartenant à un groupe est affecté à une passerelle de simulation. Douze élèves peuvent ainsi travailler simultanément comme le système supporte jusqu'à six passerelles de simulation.

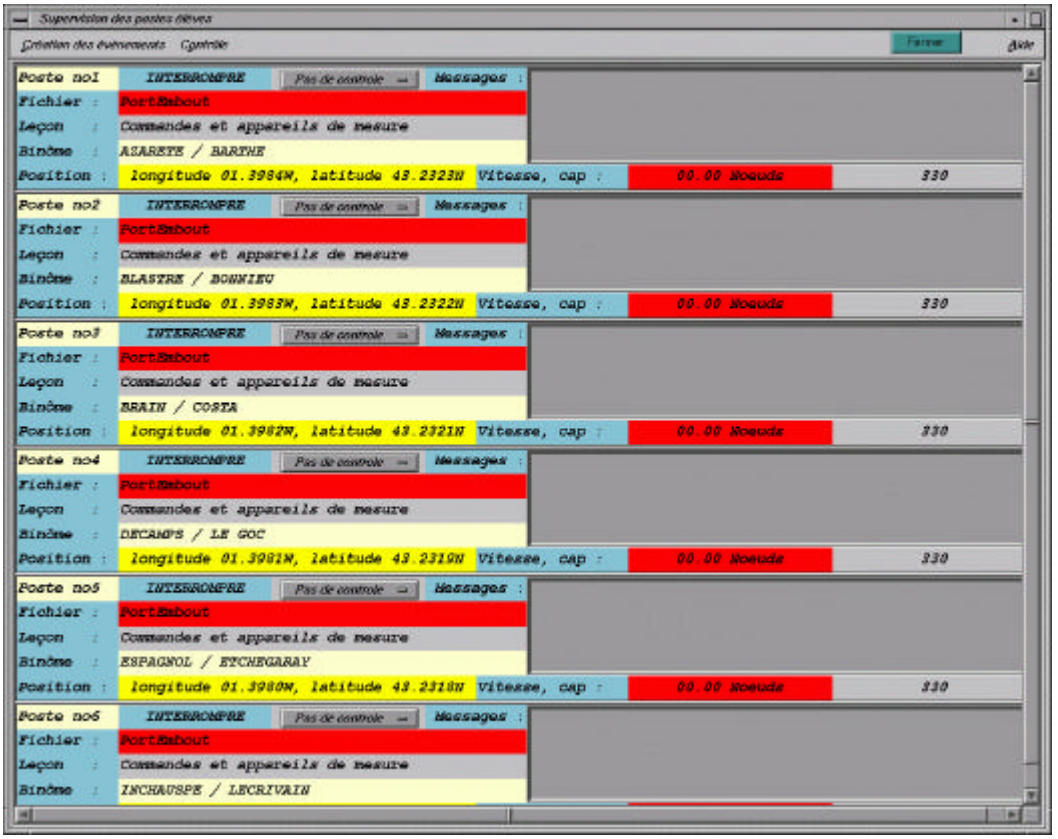


Fig. 73 : Supervision d'une session de simulation

Le simulateur a été bâti sur un principe de pédagogie active qui peut se traduire de la manière suivante. Tout d'abord, l'enseignant dispose d'un écran de supervision où lui parviennent, en temps réel, les indicateurs de base relatifs aux comportements des navires des six passerelles de simulation ainsi que des fonctions d'assistance. Il est aidé, de plus, par un système de contrôle sur chaque passerelle qui informe les élèves qu'une action doit être entreprise sans jamais en donner la solution. En cas de doute, il peut alors assurer la transmission des connaissances au moment approprié. Il peut, ainsi, consulter les images des écrans d'une passerelle sur son propre poste, décider de mettre en pause la simulation, se déplacer pour fournir des explications, puis décider de faire reprendre la simulation. Il peut également changer la configuration de l'exercice de simulation en fonction des aptitudes de chacun en introduisant, par exemple, de nouveaux navires, provoquer des pannes, rendre indisponible un instrument de navigation ou de pêche, changer l'état de la mer ou encore les conditions météo, …

Le système peut supporter jusqu'à six passerelles de simulation.

Fig. 74 : Ensemble de deux passerelles

A partir d'une étude ergonomique des postes de travail que l'on peut retrouver à bord d'un navire, différents métiers ont pu être retranscrits.

Fig. 75 : Homme de barre (au commerce) / Homme de quart ou patron (pêche)

C'est, tout d'abord, celui de l'homme de quart qui réagit aux ordres et assure la veille sur un navire de commerce. C'est le même poste pour l'homme de barre sur un navire de pêche qui manœuvre et adapte l'allure machine durant la navigation ou le patron lorsqu'il réalise les manœuvres d'accostage et d'appareillage ainsi que les manœuvres de pêche. Pour cela, la barre et les manettes de

commande sont facilement accessibles sur le poste de travail tandis que les indicateurs de bord sont affichés sur l'écran avec le paysage.

Fig. 76 : Officier de quart ou commandant (au commerce) / Patron (pêche)

C'est également celui de l'officier de quart ou du commandant sur un navire de commerce et du patron sur un navire de pêche qui gère les appareils et les cartes marines et donnent les ordres. Pour cela, une station de cartographie est mise à disposition afin de calculer la route et le cap à suivre, ou encore afficher la trace du navire, et en fonction des exercices de simulation, une aide au pointage radar pour éviter toute collision, un sonar pour traquer un banc de poissons, ou encore un netzsonde pour surveiller l'ouverture d'un chalut.

Enfin, les autres postes de travail qui se situent généralement à l'extérieur d'une passerelle de navire sont accessibles à l'aide d'un écran et d'une interface particulière pour se différencier du poste de commandement. Cela concerne les opérations d'amarrage, le poste mécanique, la gestion de l'outil de pêche, …

L'entraînement est assuré grâce à des modèles numériques reconnus et validés.

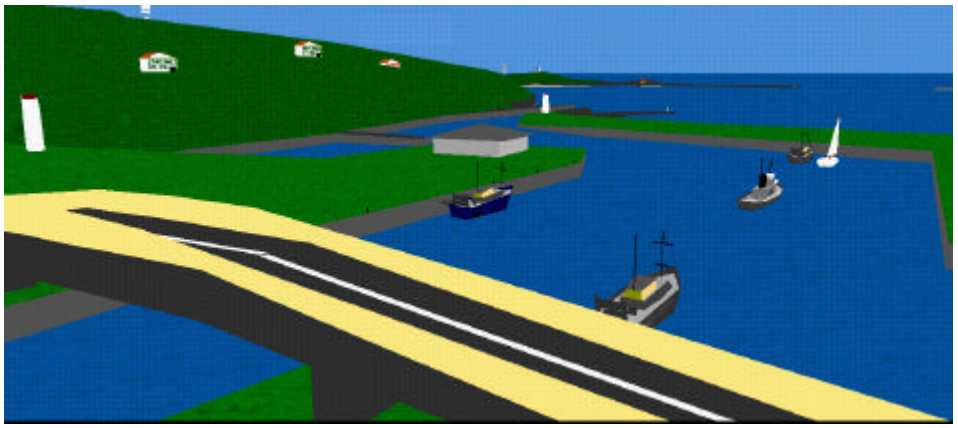


Fig. 77 : Le port de Saint-Jean de Luz

Cela est vrai pour le comportement des navires.

Fig. 78 : Vue de la passerelle

Cela est également vrai pour le comportement des filets.

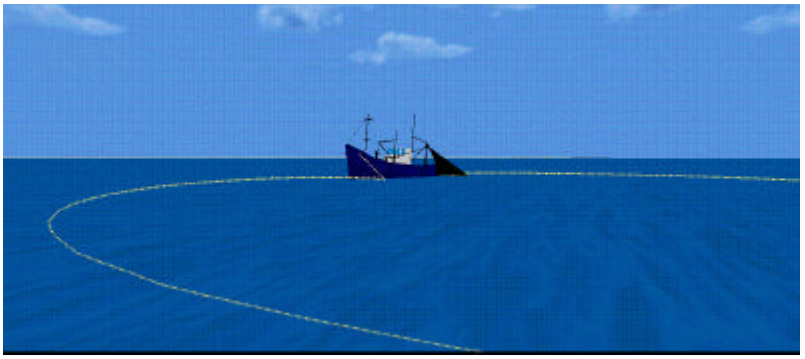


Fig. 79 : La pêche à la senne

Les instruments de navigation et de pêche sont conformes aux cahiers des charges des fabricants.

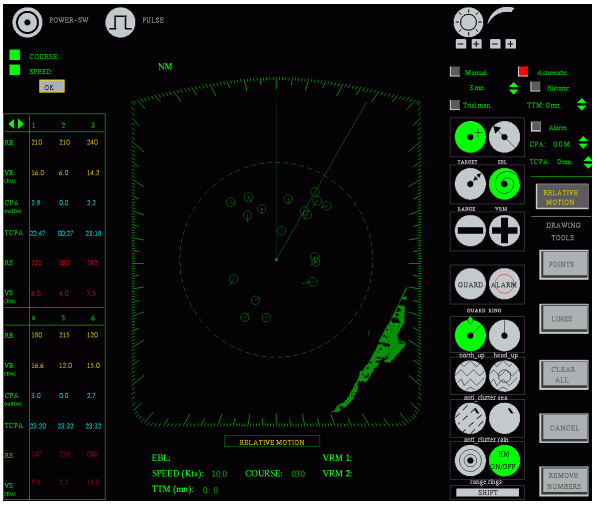


Fig. 80 : Le radar

Fig. 81 : Le sonar

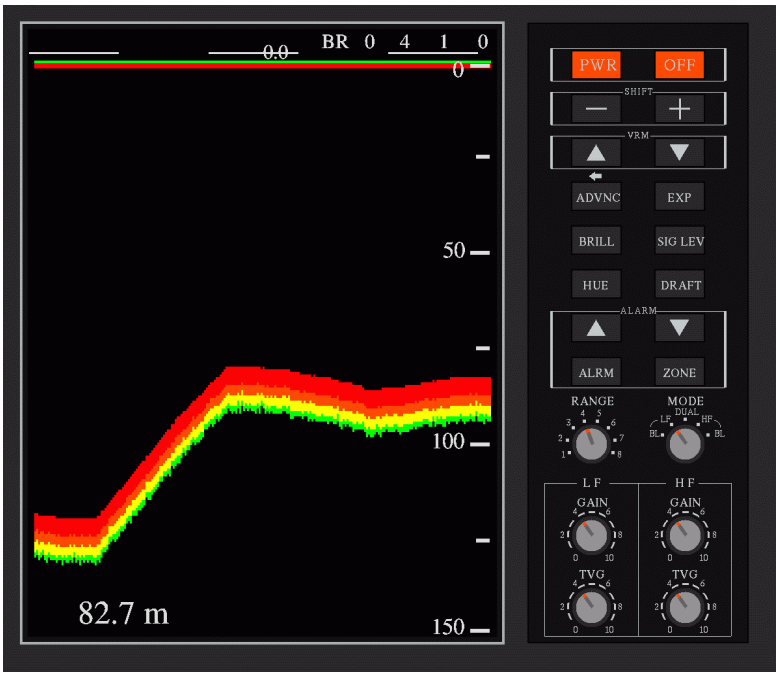


Fig. 82 : Le sondeur

L'image de synthèse permet de recréer tous les appareils que l'on trouve à bord d'un navire avec un degré d'interopérabilité maximal, ce qui permet de reconstituer des situations d'apprentissage très proches de ce qu'elles sont dans la réalité et, par là même, de former des stagiaires à avoir un comportement adéquat face à des situations qui sont toujours la conjonction de plusieurs éléments.

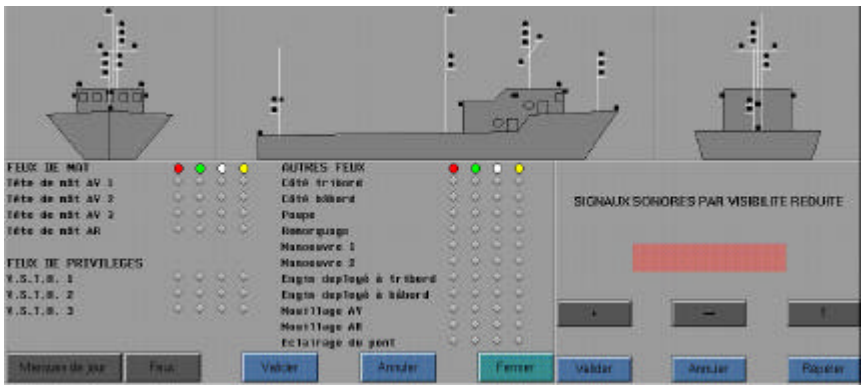


Fig. 83 : L'interface de signalisation

La signalisation est conforme au code maritime. Les cartes maritimes, publiées par le Service Hydrographique et Océanographique de la Marine (SHOM), sont affichées à l'aide de l'outil MAXSEA développé par la société Informatique & Mer.

Les valeurs ajoutées du Simulateur de Navigation et de Pêche pour un directeur d'établissement sont les suivantes : une valorisation de l'image de l'établissement, un renforcement de la professionnalisation de l'enseignement dispensé dans l'établissement, une réduction des frais de déplacement liés aux stages de simulation obligatoires effectués dans un autre établissement, une diminution des heures de conduite de bateau école et donc une réduction des frais que cela engendre, une source de revenu supplémentaire que procurent les stages de simulation dans le domaine de la formation continue.

Les valeurs ajoutées du Simulateur de Navigation et de Pêche pour l'enseignant sont les suivantes : l'utilisation d'un outil pédagogique homologué par l'Inspection Générale de l'Enseignement Maritime (IGEM), un moyen de formaliser et de mettre en pratique des cours théoriques, l'utilisation d'un outil permettant de transmettre les connaissances au moment le plus approprié, un support pédagogique permettant de se mettre au niveau de chacun, un moyen d'évaluer les connaissances d'un stagiaire.

Les valeurs ajoutées du Simulateur de Navigation et de Pêche pour le stagiaire sont les suivantes : une professionnalisation accrue de l'enseignement reçu, un apprentissage des connaissances par la pratique, un support innovant qui fait appel à la mémoire visuelle et à la mémoire auditive et qui permet d'acquérir les bons automatismes et les bons réflexes, une meilleure connaissance de l'environnement qui est le lieu de travail.

Deux types de positionnement par rapport à la concurrence sont à préciser. Par comparaison avec les simulateurs de type maquette échelle 1, traditionnellement implantés sur le marché de la simulation maritime, le Simulateur de Navigation et de Pêche propose une solution beaucoup plus interactive de l'enseignement, très proche de la réalité car l'image de synthèse permet de recréer des situations d'ensemble qui sont toujours la conjonction de plusieurs éléments avec un degré d'interopérabilité maximal. Ces situations sont toujours cohérentes et contextuelles. De plus, le coût de l'entretien est moindre puisque l'ordinateur remplace les appareils de navigation et de pêche, avec une

mise à jour en cas d'évolutions qui ne nécessite pas de changer le matériel. Par rapport aux simulateurs également basés sur une solution informatique, le Simulateur de Navigation et de Pêche se différencie quant à la complétude de la solution qu'il propose (navigation et pêche), par le fait qu'il s'appuie de plus sur une solution réseau pour favoriser un apprentissage en commun avec plusieurs navires sur un même plan d'eau, et par le principe de pédagogie active qu'il permet de mettre en application avec un poste enseignant qui supervise l'apprentissage et adapte la configuration d'un exercice en fonction des aptitudes de chacun.

ANNEXE B : GLOSSAIRE DES TERMES MARINS

Abattée : Mouvement d'un navire écartant sa route du lit du vent.

Abattre : Écarter sa route du lit du vent.

Auloffée : Mouvement d'un navire tournant son avant vers le lit du vent.

Bâbord : A gauche du navire en regardant la proue.

Bathymétrie : Science ayant pour objet la mesure des profondeurs océaniques pour déterminer la topographie du fond marin.

Beaufort : Echelle anémométrique établie en 1806 par l'amiral britannique Sir Francis Beaufort. Echelle de classification de l'état de la mer en fonction de la vitesse du vent, graduée de 0 à 12 :

Force	Appellation	Vitesse du vent (km/h)
0	Calme	1
1	Très légère brise	1 à 5
2	Légère brise	6 à 11
3	Petite brise	12 à 19
4	Jolie brise	20 à 28
5	Bonne brise	29 à 38
6	Vent frais	39 à 49
7	Grand frais	50 à 61
8	Coup de vent	62 à 74
9	Fort coup de vent	75 à 88
10	Tempête	89 à 102
11	Violente tempête	103 à 117
12	Ouragan	> 117

Carène : Partie immergée de la coque d'un navire.

Chalutier : Navire de pêche tractant un filet de forme conique sur son arrière.

Coriolis : force due à la rotation de la terre s'appliquant perpendiculairement à la trajectoire de toute particule qui s'exerce vers la droite dans l'hémisphère Nord et vers la gauche dans l'hémisphère Sud.

Etale de marée : Moment où la mer ne monte ni ne baisse.

Flux : Marée montante.

Gîte : Inclinaison latérale du navire.

Jusant : Marée descendante.

Lofer : Amener le bateau à serrer davantage le vent.

Mile marin : Unité de mesure de distance utilisée dans la marine. Un mille est égal à 1852 m, qui correspond à 1' d'angle à la surface de la terre.

Nœud : unité de mesure de vitesse utilisée dans la marine. Un nœud est égal à un mille par heure, soit 1,852 kilomètres/heure.

Poupe : Partie arrière du navire.

Proue : Partie avant du navire.

Roulis : Balancement que prend le navire dans le sens transversal.

Senneur : Navire qui pratique la pêche à la senne. Cette technique de pêche consiste à entourer un banc de poissons, puis à fermer le filet par le bas.

Tangage : Mouvement que prend le navire dans le sens longitudinal.

Tribord : À droite du navire en regardant la proue.

Turbidité : réduction de la transparence de l'eau due à la présence de particules en suspension.

Vent apparent: C'est la résultante du vent vrai et du vent dû au déplacement du bateau.

Vent vrai : Vent ressenti par un observateur lorsque le navire est à l'ancre.

Voilure : Partie émergée d'un navire.

ANNEXE C : STATE OF THE ART OF MARITIME TRAINING SIMULATOR

Deux sociétés seulement ont répondu au questionnaire présenté dans cette annexe. Ces deux sociétés ont de plus demandé que la confidentialité de leurs réponses soit respectée. C'est la raison pour laquelle nous ne communiquons pas ici leurs réponses.

I THE COAST

QI.1 : Do you use for the coast an algorithm of terrain rendering ? ☐ Yes ☐ No

QI.2 : What kind of method do you use ? (if the answer to the question Q1 was yes)

☐ A regular grid method [Ögr00]

☐ A hierarchical subdivision method [Ögr00]

QI.2.1 : Is your data structure a bintree ? [Duc97] ☐ Yes ☐ No

QI.2.2 : Is your data structure a quadtree ? [Lin96] ☐ Yes ☐ No

QI.2.3 : Another data structure

☐ A feature method [Ögr00]

QI.2.4 : Do you use the Delaunay's algorithm ? [Dev 92] ☐ Yes ☐ No

QI.2.5 : Another algorithm

☐ A refinement method [Ögr00]

QI.2.6 : Do you use a based data structure like a rectangle initially approximated by two large triangles [Hec97] Yes No ☐ ☐

QI.2.7 : Another data structure

☐ A decimation method [Ögr00]

QI.2.8 : Do You use a classical multi-resolution of level of detail ? (each resolution is pre-computed) ☐ Yes ☐ No

QI.2.9 : Do you use the View-Dependent Refinement algorithm (VDPM) ? [Hop97] ☐ Yes ☐ No

QI.2.10 : Another algorithm

☐ Another method

QI.3 : Do you automatically reproduce buildings ? ☐ Yes ☐ No

QI.4 : What kind of process do you use ? (if the answer to the question Q3 was yes)

☐ Starting from a digital camera

☐ Starting from a 3D camera

☐ Starting from another 3D optical digitizing

☐ Starting from another 3D digitizing process (which is not optical) ☐

II THE SEA

QII.1 : Do you simulate the swell ? ☐ Yes ☐ No

QII.2 : Do you take into account the effect of a local wind on the swell ? (if the answer to the question QII.1 was yes) ☐ Yes ☐ No

QII.3 : Do you simulate the wind sea in a fetch area ? ☐ Yes ☐ No

QII.3 : What kind of waves do you take into account ?

☐ Gravity waves

☐ Transition waves

☐ Capillary waves

☐ Other waves ☐

QII.4 : Do you use a dispersion relation for each kind of waves you simulate ? (if the answer to the question QII.3 was yes) ☐ Yes ☐ No

QII.5 : Do you simulate a breaking wave on the shore ? ☐ Yes ☐ No

QII.6 : Do you simulate a breaking wave in open sea ? ☐ Yes ☐ No

QII.7 : What kind of model do you use to simulate the movement of water particles ?

☐ A physical model

QII.7.1 : Do you use a finite element decomposition of Navier-Stokes equations ? [Fos96]
☐ Yes ☐ No

QII.7.2 : Others equations ☐

☐ A geometrical model based on oceanographic observations

QII.7.3 : Do you use the Gestner's geometrical model to simulate the swell ?[Cie01]
☐ Yes ☐ No Another model ☐

QII.7.4 : Do you use the Tessendorf's geometrical model to simulate the wind sea ?[Tes99]
☐ Yes ☐ No Another model ☐

QII.7.5 : What kind of wave spectrum do you use ?

☐ The wave spectrum defined by Phillips [Tes99]

☐ The wave spectrum defined by Hasselmann [Kom94]

☐ Another wave spectrum ☐

☐ Another model ☐

QII.8 : Do you take into account the wave refraction phenomenon due to the stream ?

☐ Yes ☐ No

QII.9 : Do you take into account the wave refraction phenomenon near the shore due to the sea depth ?

☐ Yes ☐ No

QII.10 : Do you take into account the wave diffraction phenomenon due to an obstacle ?

☐ Yes ☐ No

QII.11 : Do you take into account the wave reflection phenomenon due to an obstacle ?

☐ Yes ☐ No

III THE WIND

QIII.1 : Do you use a physical model to simulate the wind ?

☐ Yes ☐ No Another model ☐

IV THE STREAM

QIV.1 : What kind of streams do you take into account ?

☐ Tidal

☐ Ocean surface

☐ Other streams ☐

QIV.2 : Regarding the tidal stream, what kind of models do you use ?

☐ Hydrodynamic model [Lef00]

☐ Harmonic analysis [Lef00]

☐ Another model ☐

V THE SHIP'S BEHAVIOR

QV.1 : Regarding the ship's behavior, do you take into account the effect of :

☐ The waves

☐ The stream

☐ The wind

☐ Another effect ☐

VI THE RADAR

QVI.1 : Do you use the ray tracing algorithm to simulate the electromagnetic wave propagation ?

☐ Yes ☐ No Another algorithm ☐

QVI.2 : Do you take into account the electromagnetic wave reflection due to the waves ?

☐ Yes ☐ No Another effect ☐

VII THE SOUNDER AND THE SONAR

QVI.1 : Do you use the ray tracing algorithm to simulate the acoustic wave propagation ?

☐ Yes ☐ No Another algorithm ☐

QVII.1 : Do you take into account the acoustic wave refraction phenomenon due to the difference of water temperature ? ☐ Yes ☐ No

QVII.2 : Do you take into account the acoustic wave diffraction phenomenon due to an obstacle ?

☐ Yes ☐ No

QVII.3 : Do you take into account the acoustic wave reflection phenomenon due to an obstacle ?

☐ Yes ☐ No

ANNEXE D : COMPLEMENT SUR L'UTILISATION DE LA TRANSFORMEE DE FOURIER

Nous avons pu constater différents problèmes relatifs aux multiples implémentations du modèle de J. Tessendorf qui nous ont motivé pour rédiger ce guide d'utilisation.

Tout d'abord, nous conseillons d'utiliser la bibliothèque <rfftw.h> [fftw]. Elle a été développée au « Massachusetts Institute of Technology » (MIT) par Matteo Frigo et Steven G. Johnson. C'est une bibliothèque très performante (les tests de performance sont consultables) dont la version 3.0 est disponible depuis peu. Elle appartient à la catégorie des logiciels libres.

Il est préférable d'utiliser la bibliothèque <rfftw.h> plutôt que la bibliothèque <fftw.h> Dans le premier cas, nous définissons l'amplitude des vagues dans le domaine des fréquences pour $N^2/2 + 1$ points de la grille seulement. L'amplitude des autres points se déduit grâce à la propriété de conjugaison des nombres complexes. De plus, contrairement à la bibliothèque <fftw.h>, la partie imaginaire d'une sommation à la surface de la mer par une FFT inverse n'est pas calculée, soit le gain d'un **facteur 2**. La complexité est alors $\Theta(N/2 log_2 N)$.

On ne peut pas étudier correctement la trajectoire des particules d'eau en menant l'étude suivant l'axe des hauteurs seulement. Il faut également étudier les déplacements dans le plan de la mer, ce qui amène, à chaque pas de simulation, à calculer trois FFT inverses. L'exemple 1 suivant, pour lequel J. Tessendorf propose une solution, est ainsi incomplet. L'exemple 2 suivant comporte bien les trois FFT inverses.

L'erreur la plus manifeste que nous avons pu constater est une erreur de raisonnement. Afin d'étendre le champ des directions des vecteurs nombre d'onde dans le plan de la mer, on définit les hauteurs des vagues dans le domaine des fréquences en un nombre discrets de points d'une grille dont les bornes sont des valeurs entières comprises entre $-N/2$ et $+N/2$ (non inclus). Le point (0,0) est alors situé au milieu de la grille. Seulement, lors du calcul matriciel, les sommations sont classiquement effectuées à l'aide d'indices dont les valeurs sont comprises entre 0 et N (non inclus). Le point (0,0) est alors situé en bas et à gauche de la grille. En conséquence, il faut procéder à l'uniformisation des bornes de la sommation.

Pour y parvenir, il faut tout d'abord rendre symétrique les expressions de la transformée de Fourier discrète inverse. L'équation générale (3.35) devient alors :

$$f_{FD}(x) = \frac{1}{N} \sum_{n=-N/2}^{+N/2-1} \tilde{f}(k_n)\, e^{i k_n x} \qquad \text{(c .1)}$$

Cela a pour conséquence de diviser chacune des sommations à la surface de la mer par le nombre total de points. Or la division n'est pas effectuée par la bibliothèque <rfftw>, il faut donc l'ajouter après, c'est-à-dire diviser chacune des données de sortie par le nombre total de points.

Maintenant que nous avons exprimé les sommations entre $-N/2$ et $+N/2-1$, nous pouvons uniformiser les bornes de la sommation en nous basant sur la périodicité de chacune des fonctions. Deux approches sont alors possibles, illustrées respectivement par les exemples 1 et 2. La première approche consiste à inverser les données d'entrée (inversion par blocs de données) mais cette approche est coûteuse en temps de calcul. La deuxième approche, que nous préconisons, consiste à faire un décalage des bornes qui peut être effectué en multipliant le résultat de la sommation par –1 pour tous les indices impairs de la grille. Le lecteur pourra trouver le détail de chacune de ces deux approches dans les manuels de physique.

A titre d'illustration, nous avons ajouté un troisième exemple pour lequel les particules peuvent tourner simultanément sur des cercles et sur des ellipses.

```
// Exemple 1
for (i = 0; i < NbPoints; i ++) {
  for (j = 0; j < NbPoints; j ++) {

    // intensite du nombre d'onde
    double k = sqrt((Mer.vkx[i]*Mer.vkx[i])+(Mer.vkz[j]*Mer.vkz[j]));

    short ni = NbPoints-i;
    short nj = NbPoints-j;

    // relation de dispersion
    double wkt = sqrt (9.81 * k) * theTemps; // w(k)*t

    // Le modele de Tessendorf
    if ((i != 0) && (j != 0)) {
      complexe r (Mer.h0mdv[i][j].re, Mer.h0mdv[i][j].im);
      complexe p (cosf(wkt), sinf(wkt));

      complexe s (Mer.h0mdv[ni][nj].re, -Mer.h0mdv[ni][nj].im);
      complexe q (cosf(wkt), -sinf(wkt));

      complexe w = (r*p) + (s*q);

      Mer.inter[i][j].re = w.real ();
      Mer.inter[i][j].im = w.imag ();
    }
    else {
      Mer.inter[i][j].re = 0.0;
      Mer.inter[i][j].im = 0.0;
    }
  }
}

// Reajustement du tableau
for (i = (int)(NbPoints/2.0); i < NbPoints; i++) {
  short ni = i-((int)(NbPoints/2.0));
  for (j = (int)(NbPoints/2.0); j < NbPoints; j++) {
    short nj = j-((int)(NbPoints/2.0));

    Mer.inmdv[ni*NbPoints+nj].re = Mer.inter[i][j].re;
    Mer.inmdv[ni*NbPoints+nj].im = Mer.inter[i][j].im;
  }
  for (j = 0 ; j < (int)(NbPoints/2.0); j++) {
    short nj = ((int)(NbPoints/2.0))+j;
```

```
    Mer.inmdv[ni*NbPoints+nj].re = Mer.inter[i][j].re;
    Mer.inmdv[ni*NbPoints+nj].im = Mer.inter[i][j].im;
  }
}
for (i = 0; i < (int)(NbPoints/2.0); i++) {
  short ni = ((int)(NbPoints/2.0))+i;
  for (j = (int)(NbPoints/2.0); j < NbPoints; j++) {
    short nj = j-((int)(NbPoints/2.0));

    Mer.inmdv[ni*NbPoints+nj].re = Mer.inter[i][j].re;
    Mer.inmdv[ni*NbPoints+nj].im = Mer.inter[i][j].im;
  }
  for (j = 0 ; j < (int)(NbPoints/2.0); j++) {
    short nj = ((int)(NbPoints/2.0))+j;

    Mer.inmdv[ni*NbPoints+nj].re = Mer.inter[i][j].re;
    Mer.inmdv[ni*NbPoints+nj].im = Mer.inter[i][j].im;
  }
}

// FFT inverse
fftwnd_one (Mer.fplanmdv, &Mer.inmdv[0], 0);

// recopie du tableau
for (i = 0; i < NbPoints; i ++) {
  for (j = 0; j < NbPoints; j ++) {
    Mer.outmdv[i][j].re = Mer.inmdv[i*NbPoints+j].re/NbPoints;
    Mer.outmdv[i][j].im = Mer.inmdv[i*NbPoints+j].im/NbPoints;
  }
}
```

// Exemple 2

```
for (i = 0; i < NbPoints; i ++) {
  for (j = 0; j < (NbPoints/2+1); j ++) {

    // intensite du nombre d'onde
    double wkt;

    short ni = (NbPoints-i);
    short nj = (NbPoints-j);

    double k = sqrt((Mer.vkx[i]*Mer.vkx[i])+(Mer.vkz[j]*Mer.vkz[j]));
    double nk = sqrt((Mer.vkx[ni]*Mer.vkx[ni])+(Mer.vkz[nj]*Mer.vkz[nj]));

    // filtrage des vagues de gravité et des vagues capillaires
    if (k >= 2)
      wkt = sqrt (9.81*k*(1+k*k*0.74))*theTemps; // w(k)*t
    else
      wkt = sqrt(9.81*k)*theTemps; // w(k)*t

    if ((i != 0) && (j != 0)) {
      complexe p (cosf(wkt), -sinf(wkt));
      complexe q (cosf(wkt), sinf(wkt));
      complexe w;

      // Notre modèle : hauteurs
      w = -((Mer.h0mdv[i][j].re*p)+(Mer.h0mdv[ni][nj].re*q));

      Mer.inmdv[i*(NbPoints/2+1)+j].re = w.getRe();
      Mer.inmdv[i*(NbPoints/2+1)+j].im = w.getIm();
```

```
      if ((k != 0) && (nk != 0)) {
        complexe moinsi (0, -1);
        complexe plusi (0, 1);
        complexe x,z;

        // deplacements dans le plan
        x = (moinsi*(Mer.vkx[i]/k)*Mer.h0mdv[i][j].re*p) +
            (plusi*(Mer.vkx[ni]/nk)*Mer.h0mdv[ni][nj].re*q);
        z = (moinsi*(Mer.vkz[j]/k)*Mer.h0mdv[i][j].re*p) +
            (plusi*(Mer.vkz[nj]/nk)*Mer.h0mdv[ni][nj].re*q);

        Mer.inmdvkx[i*(NbPoints/2+1)+j].re = x.getRe();
        Mer.inmdvkx[i*(NbPoints/2+1)+j].im = x.getIm();
        Mer.inmdvkz[i*(NbPoints/2+1)+j].re = z.getRe();
        Mer.inmdvkz[i*(NbPoints/2+1)+j].im = z.getIm();
      }
      else {
        Mer.inmdvkx[i*(NbPoints/2+1)+j].re = 0.0;
        Mer.inmdvkx[i*(NbPoints/2+1)+j].im = 0.0;
        Mer.inmdvkz[i*(NbPoints/2+1)+j].re = 0.0;
        Mer.inmdvkz[i*(NbPoints/2+1)+j].im = 0.0;
      }
    }
    else {
      Mer.inmdv[i*(NbPoints/2+1)+j].re = 0.0;
      Mer.inmdv[i*(NbPoints/2+1)+j].im = 0.0;
      Mer.inmdvkx[i*(NbPoints/2+1)+j].re = 0.0;
      Mer.inmdvkx[i*(NbPoints/2+1)+j].im = 0.0;
      Mer.inmdvkz[i*(NbPoints/2+1)+j].re = 0.0;
      Mer.inmdvkz[i*(NbPoints/2+1)+j].im = 0.0;
    }
  }
}

// FFT inverse
rfftwnd_one_complex_to_real(Mer.fplanmdv, &Mer.inmdv[0], &Mer.inoutmdv[0]);
rfftwnd_one_complex_to_real(Mer.fplanmdvkx,
      &Mer.inmdvkx[0],Mer.inoutmdvkx[0]);
rfftwnd_one_complex_to_real(Mer.fplanmdvkz,
      &Mer.inmdvkz[0],  &Mer.inoutmdvkz[0]);

// Reajustement du tableau : 2 eme methode
// recopie du tableau
for (i = 0; i < NbPoints; i ++) {
  for (j = 0; j < NbPoints; j ++) {
    Mer.outmdv[i][j] = pow(-1,i+j) * Mer.inoutmdv[i*NbPoints+j] / NbPoints;
    Mer.outmdvkx[i][j] = pow(-1,i+j)*Mer.inoutmdvkx[i*NbPoints+j]/NbPoints;
    Mer.outmdvkz[i][j] = pow(-1,i+j)*Mer.inoutmdvkz[i*NbPoints+j]/NbPoints;
  }
}
```

// Exemple 3

```
for (i = 0; i < NbPoints; i ++) {
  for (j = 0; j < NbPoints; j ++) {
    // intensite du nombre d'onde

    double wkt;
    short ni = NbPoints-i;
    short nj = NbPoints-j;
```

```
double k = sqrt((Mer.vkx[i]*Mer.vkx[i])+(Mer.vkz[j]*Mer.vkz[j]));
double nk = sqrt((Mer.vkx[ni]*Mer.vkx[ni])+(Mer.vkz[nj]*Mer.vkz[nj]));

// filtrage des vagues de gravité et des vagues capillaires
if (k >= 2.0)
  wkt = sqrt (9.81*k*(1+k*k*0.074))*theTemps; // w(k)*t
else
  wkt = sqrt(9.81*k)*theTemps; // w(k)*t

static float Sx = 1/(1-exp(-0.11*profondm));
static float Sz = Sx*(1-exp(-0.09*profondm));
static float taux = abs(sinf(pentefond)*exp(-0.1*profondm);
static float tauxprime = sqrt(1-taux);

if ((i != 0) && (j != 0)) {
  // filtrage des vagues de gravité et des vagues capillaires
  if (k < 2.0) {
     complexe p (cosf(wkt), -sinf(wkt));
     complexe q (cosf(wkt), sinf(wkt));
     complexe r1 (tauxprime*Sz*Mer.h0mdv[i][j].re,
        taux*Sx*Mer.h0mdv[i][j].re);
     complexe s1etoile (tauxprime*Sz*Mer.h0mdv[ni][nj].re, -
        taux*Sx*Mer.h0mdv[ni][nj].re);

     complexe w;
     w = -(r1*p+s1etoile*q);

     Mer.inmdv[i*NbPoints+j].re = w.getRe();
     Mer.inmdv[i*NbPoints+j].im = w.getIm();

    if ((k != 0) && (nk != 0)) {
      complexe r2 (tauxprime*Sx*Mer.h0mdv[i][j].re,
        taux*Sz*Mer.h0mdv[i][j].re);
      complexe s2etoile (tauxprime*Sx*Mer.h0mdv[ni][nj].re, -
        taux*Sz*Mer.h0mdv[ni][nj].re);
      complexe moinsi (0, -1);
      complexe plusi (0, 1);
      complexe x, z;

      x = (moinsi*(Mer.vkx[i]/k)*r2*p) +
        (plusi*(Mer.vkx[ni]/nk)*s2etoile*q);
      z = (moinsi*(Mer.vkz[j]/k)*r2*p) +
        (plusi*(Mer.vkz[nj]/nk)*s2etoile*q);

      Mer.inmdvkx[i*NbPoints+j].re = x.getRe();
      Mer.inmdvkx[i*NbPoints+j].im = x.getIm();
      Mer.inmdvkz[i*NbPoints+j].re = z.getRe();
      Mer.inmdvkz[i*NbPoints+j].im = z.getIm();
   }
   else {
     Mer.inmdvkx[i*NbPoints+j].re = 0.0;
     Mer.inmdvkx[i*NbPoints+j].im = 0.0;
     Mer.inmdvkz[i*NbPoints+j].re = 0.0;
     Mer.inmdvkz[i*NbPoints+j].im = 0.0;
   }
 }
 else {
   complexe p (cosf(wkt), -sinf(wkt));
   complexe q (cosf(wkt), sinf(wkt));
   complexe w;
```

```
        w = -((Mer.h0mdv[i][j].re*p)+(Mer.h0mdv[ni][nj].re*q));

        Mer.inmdv[i*NbPoints+j].re = w.getRe();
        Mer.inmdv[i*NbPoints+j].im = w.getIm();

        complexe moinsi (0, -1);
        complexe plusi (0, 1);
        complexe x,z;

        x = (moinsi*(Mer.vkx[i]/k)*Mer.h0mdv[i][j].re*p) +
             (plusi*(Mer.vkx[ni]/nk)*Mer.h0mdv[ni][nj].re*q);
        z = (moinsi*(Mer.vkz[j]/k)*Mer.h0mdv[i][j].re*p) +
             (plusi*(Mer.vkz[nj]/nk)*Mer.h0mdv[ni][nj].re*q);

        Mer.inmdvkx[i*NbPoints+j].re = x.getRe();
        Mer.inmdvkx[i*NbPoints+j].im = x.getIm();
        Mer.inmdvkz[i*NbPoints+j].re = z.getRe();
        Mer.inmdvkz[i*NbPoints+j].im = z.getIm();
      }
    }
    else {
      Mer.inmdv[i*NbPoints+j].re = 0.0;
      Mer.inmdv[i*NbPoints+j].im = 0.0;
      Mer.inmdvkx[i*NbPoints+j].re = 0.0;
      Mer.inmdvkx[i*NbPoints+j].im = 0.0;
      Mer.inmdvkz[i*NbPoints+j].re = 0.0;
      Mer.inmdvkz[i*NbPoints+j].im = 0.0;
     }
   }
}

// FFT inverse
fftwnd_one (Mer.fplanmdv, &Mer.inmdv[0], 0);
fftwnd_one (Mer.fplanmdvkx, &Mer.inmdvkx[0], 0);
fftwnd_one (Mer.fplanmdvkz, &Mer.inmdvkz[0], 0);

// Reajustement du tableau : 2 eme methode
// recopie du tableau
for (i = 0; i < NbPoints; i ++) {
  for (j = 0; j < NbPoints; j ++) {
    Mer.outmdv[i][j].re = powf(-1,i+j)*Mer.inmdv[i*NbPoints+j].re/NbPoints;
    Mer.outmdvkx[i][j].re = powf(-1,i+j)*Mer.inmdvkx[i*NbPoints+j].re/
        NbPoints;
    Mer.outmdvkz[i][j].re = powf(-1,i+j)*Mer.inmdvkz[i*NbPoints+j].re/
        NbPoints;
    Mer.outmdvkz[i][j].re = powf(-1,i+j)*Mer.inmdvkz[i*NbPoints+j].re/;
        NbPoints;
  }
}
```

ANNEXE E : UN EXEMPLE DE VERTEX SHADER ET DE PIXEL SHADER

Nous présentons, ici, seulement un exemple de « vertex shader » et de « pixel shader » parmi tous ceux que nous avons écrits.

```
// VERTEX SHADER

struct inputs
{
   float4 Position     : POSITION;
   float4 Normal       : NORMAL;
};

struct outputs
{
   float4 hPosition    : POSITION;
   float3 coulsolciel  : COLOR0;
   float3 couleau      : TEXCOORD0;
   float4 reflectVec   : TEXCOORD1;
   float4 refractVec   : TEXCOORD2;
   float  fresnelTerm  : TEXCOORD3;
};

outputs main(inputs IN,
          uniform float4x4 ModelViewProj,
          uniform float4x4 ModelView,
          uniform float4x4 ModelViewIT,
               uniform float4 Eye,
               uniform float4 Sol)
{
   outputs OUT;

   // position
   OUT.hPosition = mul(ModelViewProj, IN.Position);

   float3 eaurgb = float3(0.05, 0.13, 0.35);
   float3 cielrgb = float3(0.35, 0.65, 0.88);
   float3 soleilrgb = float3(1.0, 0.6, 0.25);

   // thetai
   float3 normal = IN.Normal.xyz;
   normal = normalize(normal);
   float3 PosToEye = Eye.xyz - IN.Position.xyz;
   PosToEye = normalize(PosToEye);
   float cosine = dot(PosToEye, normal);

   // fresnel term
   float fresnel = pow(1/(1.0 + cosine), 8);
   OUT.fresnelTerm = fresnel;
   float mfresnel = (1 - fresnel);

   // reflectvec
   float3 Reflect = (2*cosine*normal) - PosToEye;
   Reflect = normalize(Reflect);
   OUT.reflectVec.xyz = Reflect;
   OUT.reflectVec.w = 1;
```

```
    // refractvec
    float sine = 1-cosine*cosine;
    float costerm = pow(1-sine*1.777, 0.5);
    float3 Refract;
    if (cosine < 0) {
      Refract.x = (normal.x*(cosine*1.333 + costerm)) - (PosToEye.x*1.333);
      Refract.y = (normal.y*(cosine*1.333 + costerm)) - (PosToEye.y*1.333);
      Refract.z = (normal.z*(cosine*1.333 + costerm)) - (PosToEye.z*1.333);
    }
    else {
      Refract.x = (normal.x*(cosine*1.333 - costerm)) - (PosToEye.x*1.333);
      Refract.y = (normal.y*(cosine*1.333 - costerm)) - (PosToEye.y*1.333);
      Refract.z = (normal.z*(cosine*1.333 - costerm)) - (PosToEye.z*1.333);
    }
    Refract = normalize(Refract);
    OUT.refractVec.xyz = Refract;
    OUT.refractVec.w = 1;

    // sun and sky color
    float cosreflectsol = dot(Reflect, Sol.xyz);

    if ((cosreflectsol > 0.95) && (fresnel > 0.6)) {
      OUT.coulsolciel.r = (pow(cosreflectsol,10)*soleilrgb.x);
      OUT.coulsolciel.g = (pow(cosreflectsol,10)*soleilrgb.y);
      OUT.coulsolciel.b = (pow(cosreflectsol,10)*soleilrgb.z);
    }
    else {
      OUT.coulsolciel.r = 0.0;
      OUT.coulsolciel.g = 0.0;
      OUT.coulsolciel.b = 0.0;
    }

    // water and sky color
    float sinthetat = pow(sine, 0.5)*1.333;
    OUT.couleau.r = (eaurgb.x + sinthetat*0.09);
    OUT.couleau.g = (eaurgb.y + sinthetat*0.25);
    OUT.couleau.b = (eaurgb.z + sinthetat*0.23);

    return OUT;
}


// PIXEL SHADER

float4 main(in float3 coulsolciel  : COLOR0,
            in float3 couleau      : TEXCOORD0,
            in float4 reflectVec   : TEXCOORD1,
            in float4 refractVec   : TEXCOORD2,
            in float fresnelTerm   : TEXCOORD3,
            uniform samplerCUBE MapsCiel,
            uniform samplerCUBE MapsFond) : COLOR
{
    float3 cielrgb = float3(0.35, 0.65, 0.88);
    float3 reflectColor = fresnelTerm*(0.5*texCUBE(MapsCiel, reflectVec).rgb + 0.5*cielrgb.rgb);
    float mfresnelTerm = (1 - fresnelTerm);
    float3 refractColor = mfresnelTerm*(0.3*texCUBE(MapsFond, refractVec).rgb + 0.7*couleau);

    return float4(reflectColor+refractColor+coulsolciel, 1.0);
}
```

BIBLIOGRAPHIE

[AA85] C. Aristaghes and P. Aristaghes. *Théories de la houle, houle réelle, propagation de la houle.* Services techniques central des ports maritimes et voies navigables, 1985

[AP90] R.W. Austin, T. Petzold. *Spectral Dependence of the Diffuse Attenuation Coefficient of Lighting Ocean Waters : A Re-examination Using New Data.* Ocean Optics X, Richard W. Spinrad, ed., SPIE 1302, 1990

[ATI] http:www.ati.com

[Bec00] Beck R.F. and Reed A.M. *Modern seakeeping computations for ships.* 23nd Symposium on Naval Hydrodynamics, Bassin des Carènes, Val de Reuil, France, 2000

[Bie52] F. Biesel. *Study of wave propagation in water gradually varying depth*. In Gravity waves, p 243-253, U.S. National Bureau of Standards Circular 521, 1952

[Bli77] J.F. Blinn. *Modelsof light refection for computer synthesized pictures.* In Siggraph'77, p 192-198, 1977

[Blo00] J. Blow. *Terrain Rendering at High Levels of Detail.* proceedings of the 2000 Game Developers Conference, March 2000

[BG69] F.P. Bretherton and C.J.R. Garett. *Wavetrains in in homogeneous moving media.* Proceedings of R. Soc. London, Ser. A, 302, p 529-554, 1969

[Caz97] F. Cazals. *Structures de Données Hiérarchiques non Récursives et Problèmes de Proximié.* Thèse de l'université de Paris VII, 1997

[CL95] J. Chen and N. Lobo. *Toward interactive-rate simulation of fluids with moving obstacles using navier-stokes equations.* In graphical models and Image processing, p 107-116, March 1995

[Cie01] J.M. Cieutat, J.Ch. Gonzato, P. Guitton. *A new efficient wave model for maritime training simulator*, in Spring Conference on Computer Graphics (éd. Tosiyasu L. Kunii), IEEE Computer Society (2001), 251-259

[Cie02-1] J.M. Cieutat, J.C. Gonzato, P. Guitton. *Wave generation and propagation for maritime training simulator.* 6th world multi-conference on Systemics, Cybernetics and Informatics, Orlando, Florida, USA, July 14-18, 2002, Proceedings volume XII, International Institute of Informatics and Systemics (IIIS), p 55-62

[Cie02-2] J.M. Cieutat, J.C. Gonzato, P. Guitton. *Navigation training simulation in ocean waves.* Sim'Ouest'2002, European Conference on marine Technology : Industry and Simulation, November 28-29 2002

[Cie03] J.M. Cieutat, J.C. Gonzato, P. Guitton. *A general ocean waves model for ship design*.Virtual Concept'2003, France, November 2003

[Cg] http://www.nvidia.com/object/cg_toolkit.html

[CL95] J.Chen, N. Lobo. *Toward interactive-rate simulation of fluids moving obstacles using navier-stokes equations.* In Graphical Models and Image Processing, p 107-116, March 1995.

[CNRM] http:www.meteo.fr

[Com90] R. Comolet. *Mécanique des fluides.* Masson, 1990

[Den89] D.B. Creamer, F. Henyey, R. Schult, J. Wright. *Improved Linear Representation of Ocean Surface Waves.* J. Fluid Mech., 205, p 135-161, 1989

[Duc97] M. A. Duchaineau, M. Wolinsky, D.E. Sigeti, M.C. Miller, C. Aldrich, and M.B. Mineev-Weinstein. *ROAMing Terrain : Real-time Optimally Adaptating Meshes.* IEEE Visualisation'97, p 81-88. November 1997

[EnmmM] Enmm-marseille@wanadoo.fr

[EnmmN] Enmm.Nantes@wanadoo.fr

[EnmmS] Enmm.Saint-malo@wanadoo.fr

[Estia] htpp://www.estia.fr

[Eva97] W. Evans, S. S. Kiena, A. Varshney. *Optimizing Triangle Strips for Fast Rendering.* IEEE Visualization'96, p 319-326, October 1996

[Eva01] W. Evans, D. Kirkpatrick, and G. Townsend. *Right-triangulated Irregular Networks*. Algorithmica : Special Issue on Algorithms for Geographical Information, 30(2) (2001) p 264-286

[Faros] htpp://www.faros.com

[fftw] htpp://www.fftw.org

[Fer03] R. Fernando & M. Kilgard. *The Cg tutorial. The Definitive Guide to Programmable Real-Time Graphics.* Addison-Wesley, 2003

[Fle91] C.A.J. Fletcher. *Computational techniques for fluid dynamics.* vol. 2. Springe series in computational techniques, 1991

[Flo93] L. Floriani, P. Magillo. *Algorithms for Visibility Computation on Digital terrain Models.*ACM 1993

[Fol99] Foley, Van Dam, Freiner, Hughes. *Computer Graphics, principles and practice*. Second edition. Addison Wesley, November 1999

[FM96] N. Foster and D. Metaxas. *Realistic animation of liquids.* In Graphics Interface'96, 1996

[FM97] N. Foster and D. Metaxas. *Controlling fluid animation. CGI'97*, p 178-188, 1997

[FR86] A. Fournier and W. Reeves. *A simple model of ocean waves*. In SIGGRAPH'86, volume 20, p 75-84, 1986

[Fun93] T. Funkhouser, C. Séquin. *Adaptive display algorithm for interactive frame rates durin visualization of complex virtual environments.* Computer Graphics (SIGGRAPH'93 Proceedings), p 247-254, 1993

[Gamas] htpp://www.gamasutra.com

[Ger09] F.J. Gerstner. *Theorie der wellen.* (reprint) Ann des Physik, 32 : p 412-440, (1802) 1809

[Gla88] A.S. Glassner. *How to derive a spectrum from a rgb triplet.* Computer Graphics and Applications, 8(3): p 60-70, 1978

[Gol01] R. Golias, L.S. Jensen. *Deep-Water Animation and Rendering*. GDCE 2001

[Gon97] J.C. Gonzato and B. Le Saëc. *A phenomenological model of coastal scenes based on physical considerations*. In Springer-Verlag, editor, 8th Eurographics Workshop on Computer Animation and Simulation, p 137-148, 1997

[Gon99] J.C. Gonzato. *Modelisation des scènes océaniques*. PhD thesis, LaBRI, Université Bordeaux I, Décembre 1999

[Gon00]J.C. Gonzato, B. Le Saëc. *On modelling and rendering ocean scenes*. Journal of Vizualization and Computer Animation, 11(1) : p 27-37, 2000

[GGS95] Markus H. Gross, Roger Gatti, and Oliver G. Staadt. *Fast Multiresolution Surface Meshing*. In Visualization *95*. IEEE, October 1995

[Hal89] R. Hall. *Illumination and Color for Computer Generale Imagery.* Springer-Verlag, 1989

[HW80] Haltiner, G. J., and R. T. Williams. *Numerical Weather Prediction and Dynamic meteorology*. Wiley, New York, p 477, 1980

[Has94] G.L. Komen,, L. Caveleri, M. Donelan, K. Hasselmann, S. Hasselmann, and M. Janssen (Eds.), 1994: *Dynamics and Modelling of Ocean Waves,* Cambridge

[Hav13] T.H. Havelock. *Sip Resistance: The Wave-making Properties of Ceratain travelling Pressure Disturbances.* Proceedings of the Royal Society, Vol. A 89, p 490-499, 1913

[Hec94] P.S. Heckbert, M. Garland. *Multiresolution Modeling for fast Rendering.* In proceedings of Graphics Interface'94, p 1-8, 1994

[Hop97] H. Hoppe. *Smooth View-Dependent Refinement of Progressive Meshes.* Proceeedings of SIGGRAPH'97, p 189-198, August 1997

[Hop98] H. Hoppe. *Smooth View-Dependent Level-of-detail Control and Application to Terrain Rendering.* IEEE Visualization'98, p 35-42, October 1998

[Ifrem] http://www.ifremer.fr

[Ima95] A. Imamiya, D. Zhang. *Modelling breaking ocean waves, influence of floor and refraction.* In Pacific Graphics 95, 1995

[IM] http://www.maxsea.com

[Intel] http://www.intel.com

[ISO9126] ISO 9126. Information technology. Software product evaluation. Quality characteristics and guidelines for their use, 1994

[Jer68] N.G. Jerlov. *Optical Oceanography*, Elsevier, Amsterdam, 1968

[Jon90] Jonsson I. G.,1990. *Wave-current interactions.* In: Le Mehauté, B. Hanes, D.M. (Eds), 'The Sea'. Chap. 3, Vol. 9, Part A

[Kin65] B. Kinsman. *Wind Waves, their generation and propagation on the ocean surface.* Prentice-Hall, Inc., Englewood Cliffs, New jersey, 1965

[KM90] M.Kass, G. Miller. *Rapid, stable fluid dynamics forfor computer graphics.* Comput. Graph., vol. 24, p 49-55, 1990

[Kongs] http://www.kongsberg.com

[Kro55] B.V. Korvin-Kroukovsky. *Investigations of Ship Motions in Regular Waves.*trans. SNAME, Vol. 63, p 386-435, 1955

[Lac65] H. Lacombes. *Cours d'Océanographie Physique.* Gauthier-Villars, Paris, 1965

[Lay02] Anita T. Layton, Michiel van de Panne. *A numerically efficient and stable algorithm for animating water waves*. The Visaul Computer, vo1 18, p 41-53, 2002

[Mic98] Michell J.H. *Wave resistance of a Ship.* Philosophical Magazine, Vol 45, p 113, London, 1898

[Mon86] E.C. Monahan, G. Mac Niocaill, editors. *OceanicWhitecaps : Their Role in Air Sea Exchange Processes.* Dordrecht, reidel, 1986

[Lac65] H. Lacombes. *Cours d'océanographie physique*. Gauthier-Villars, Paris, 1965

[Lef00] F. Lefevre. *Modélisation des marées océaniques à l'échelle globale : assimilation de données in situ et altimétriques.* PhD thesis, Université Toulouse III, 22 septembre 2000

[Lin96] P. Lindstrom, D. Koller,, W. Ribarsky, L.F. Hodges, N. Faust, and G. Turner. *Real-Time, Continuous Level of detail Rendering of Heights Fields*. Proceedings of SIGGRAPH'96, p 109-118, August 1996

[Lin97] P. Lindstrom, D. Koller,, W. Ribarsky, L.F. Hodges, N. Faust, and G. Turner. *An Integrated Global GIS and Visual Simulation System.* tech. Rep. GIT-GVU-97-07, Georgia Institute of Technology, March 1997

[Mas87] G.A. Mastin, P.A. Watterger and J.F. Mareda. *Fourier synthesis of ocean scenes*. IEEE Computer Graphics and Applications, p 16-23, March 1987

[Ney1] D. Hinsinger, F. Neyret, M.P. Cani. *Interactive Animation of ocean Waves.* Composium Computer Animation'02. San Antonio, July 2002

[NVI] http://www.nvidia.com

[OGL96] M. Woo, J. Neider, T. Davis. *OpenGL Programming Guide.* Second edition. The Official Guide to Learning OpenGL, Version 1.1. Addison Wesley Developers Pres, 1996.

[OInv] http://www.sgi.com/software/inventor

[OpSG] http://www.sense8.com

[Ögr00] Andreas Ögren. *Continuous Level Of Detail in Real-Time Terrain Rendering.* Master's Thesis, university of Umea, Sweden, Januar 2000

[OT69] Ogilvie, T.F.& E.O. Tuck. *A Rational Strip theory of Ship Motions.* Part 1, Report 013, Department of Naval Architecture, University of Michigan, 1969

[Paj98] R.B. Pajarola. *Large Scale Terrain Visualization Using the Restricted Quatree triangulation*. IEEE Visualization'98, p 19-26, October 1998

[Pas01] V. Pascucci, R.J. Frank. *Global Static Indexing for Real-time Exploration of Very Large Regular Grids.* Proceedings of Supercomputing, November 2001

[Pea86] Darwyn peachey. *Modeling Waves and Surf.* Computer Graphics, vol. 20, no. 4, 1986, p 65-74

[Ped00] T. Pedersen. *Wave Load Prediction - a Design Tool.* PhD thesis, January 2000. Department of Naval Architecture

[Perfor] http://www.sgi.com/software/performer

[Phi57] O.M. Phillips. *On the generation of waves by turbulent wind.* Journal of Fluid Mechanics, vol. 2, p 417-445, 1957

[Phi77] O.M. Phillips. *The dynamics of the Upper Ocean.* Cambridge University Press, Cambridge, 1977

[P&M64] W.J. Pierson and L. Moskowitz. *A proposed spectral form for fully developped wind seas based on similarity theory of S.A. Kilaigorodskii.* Jpurnal of Geophysical Researc, p 5181-5190, 1964

[Pre00] S. Premoze, M. Ashikhmin. *Rendering Natural Waters*. The proceedings of Pacific Graphics 200, Hong-Kong, October 3-5, p 23-30

[Pho75] B.T. Phong. *Illumination for computer generated pictures*. CACM, 18 : p 311-317, 1975

[Rab97] B. Rabinovich, C. Gotsman. *Visualization of Large Tterrains in Resource-Limited Computing Environments.*IEEE Visualization'97, p 95-102, November1997

[Ran63] W.J.M. Rankine. *On the exact form of waves near the surface of deep water.* Phil. Trans. Roy. Soc. A, 153: p 127-138, 1863

[Red99] M. Reddy, Y. Leclerc, L. Iverson, N. Bletter. Terra Vision II. *Visualizing Massive Terrain Databases in VRML.* IEEE Computer Graphics & Applications, p 30-38, April 1999

[Roh94] J. Rohlf, J. Helman. *IRIS-Performer : A high Performance Multiprocessing Toolkit for real-Time 3D Graphics.* Proceedings of SIGGRAPH 94, p 381-395, July 1994

[Röt98] S. Röttger,W. Heidrich, P. Slussallek, P.H. Seidel. *Real-Time Generation of Continuous Levels of Detail for Height Fields.* Proceedings of the 6th International Conference in central Europe on Computer Graphics and Visualization, p 315-322, February 1998

[Rus00] S. Rusinkiewicz, M. Levoy. *QSPLAT : A Multiresolution Point Rendering System for Large Meshes.* Proceedings of SIGGRAPH 2000, p 343-352, July 2000

[Sam84] H. Samet. *The quadtree and related hierarchical data structures.*In Computer Surveys, vol. 16(2), p 187-260, 1984

[SGI] http://www.sgi.com

[STF70] Salvesen N., Tuck E.O. and Faltinsen O. *Ship Motions and Sea Loads*. Transactions of the Society of Naval Architects and marine Engineers, vol. 78, p 250-287, 1970

[Star01] Joe Starm. *A Simple Fluid Solver based on the FFT*. Journal of Graphics Tools. Available at http://www.acm.org/jgt/papers/Starm01

[STCW95] STCW. *Normes de formation des gens de mer, de délivrance des brevets et de veille.* Convention STCW du 26 juin au 7 juillet de l'Organisation maritime Internationale

[Tes01-1] J. Tessendorf. *Simulating ocean water*. In SIGGRAPH'2001, course notes

[Tes01-2] J. Tessendorf. *Simulating ocean water*. In SIGGRAPH'2001, slides

[TB87] P.Y. Ts'o and B.A. Barsky. *Modellling and rendering waves : Wave-tracing using beta-splines and reflective and refractive texture mapping.* In ACM Transactions on Graphics, volume 6, p 191-214, July 1987

[Trans] http://www.transas.com

[Vit01] J.S. Vitter. *External Memory Algorithms and data Structures : Dealing with MASSIVE DATA.* ACM Computing survey, 2001

[VTP] http://www.vterrain.org/Water/index.htm

[Wat90] M. Watt. *Light-Water Interaction using backward Beam tracing.* Proceedings of SIGGRAPH'90, in Computer Graphics 24, p 377-385

[Wat99] A. Watt, F. Policarpo. *The Computer Image.* Édition Addison-Wesley, 1999

[Wol99] J. Wolf, D. Prandle. *Some observations of wave-current interactions.* Coastal Engineering, 37(3-4): p 471-485

[WldTk] http://www.sense8.com

[Xu97] Y. Xu and al. *Physically based simulation of water currents and waves.* Computer & Graphics, 21 : p 277-280, 1997

Résumé :

La simulation d'entraînement maritime est une matière importante de l'enseignement maritime qui fait appel à de nombreuses disciplines scientifiques et techniques.

Dans ce contexte où la contrainte de temps réel doit être satisfaite, l'ensemble des phénomènes physiques ne peut être traité ; seuls les phénomènes physiques les plus visuels liés aux éléments naturels et au comportement du navire sont alors représentés.

Notre modèle de houle, basé sur une approche de simulation d'une onde de surface, permet de simuler la forme et la propagation d'un train régulier de vagues depuis la pleine mer jusqu'au bord du rivage tout en tenant compte des effets du courant et de la profondeur. Notre modèle de mer du vent, plus limitatif parce que basé sur une approche de simulation spectrale, est très représentatif des états de mer qui sont définis sur l'échelle de Beaufort allant de 0 à 12 en fonction des conditions de vent.

Le choix d'une structure de données multi-grille à échelle fixe, l'affichage par niveaux de détails, l'ajout de détails apparents pour les vagues capillaires à l'aide de la technique de «bump mapping » nous permettent d'intégrer toutes les échelles, de la ride millimétrique à un golfe dans son ensemble, au sein d'un même outil interactif animé. Concernant le modèle d'éclairage, nous tirons bénéfice des facilités offertes par les cartes graphiques programmables et une amélioration du calcul temps réel de la couleur retransmise par la masse d'eau est également proposée.

Abstract :

Maritime training simulation is an important matter of maritime teaching, which requires a lot of scientific and technical skills.

In this framework, where the real time constraint has to be maintained, all physical phenomena cannot be studied; the most visual physical phenomena relating to the natural elements and the ship behaviour are reproduced only.

Our swell model, based on a surface wave simulation approach, permits to simulate the shape and the propagation of a regular train of waves from the open sea to the shore taking into account current and depth effects. Our wind sea model is more restricted because it is based on a spectral simulation approach but the obtained results are very representative of the sea states, which are defined on the Beaufort's scale from 0 to 12 depending on the wind conditions.

The choice of multi-grid data structure with a fixed scale, level of details display, bump-mapping of capillary waves permit to integrate all scales, from a millimetric wave amplitude to a whole golfe, inside a unique interactive animated tool. Relating the lighting model, we use vertex and pixel shaders and an improved real-time computation of the underwater color is also proposed.