

Modélisation de l'Hydroptère

Basile Graf

18 juin 2006

Buts du projet

Introduction

Géométrie du modèle

Génération du modèle

Simulation et optimisation

Simulation

Optimisation

Intégration

Performances

Réalité virtuelle

Réalité virtuelle

Programme

Conclusion

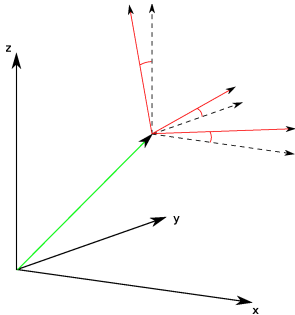


Buts

- ▶ Optimiser le modèle de l'hydroptère
 - ▶ Les temps de calculs sont trop importants
- ▶ Traduction (semi-automatique) du modèle de MATLAB à C++
- ▶ Implémentation en temps réel
- ▶ Réalité virtuelle
 - ▶ Affichage 3D
 - ▶ Pilotage
- ▶ (Documentation) (!)

Géométrie du modèle

On considère le bateau comme un solide



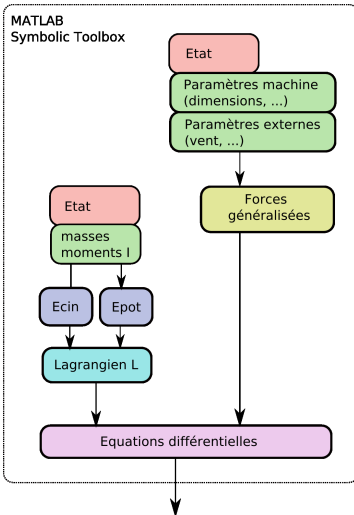
Position et attitude

$$\vec{r} = \begin{pmatrix} x \\ y \\ z \end{pmatrix} \quad \vec{a} = \begin{pmatrix} \phi \\ \theta \\ \psi \end{pmatrix}$$

Le vecteur d'état est donc

$$\vec{q} = \begin{pmatrix} \vec{r} \\ \vec{a} \\ \vec{r} \\ \vec{a} \end{pmatrix} \quad \dim(\vec{q}) = 12$$

Génération du modèle sous MATLAB



- Évolution décrite par

$$\dot{\vec{q}} = \vec{f}(\vec{q}, \vec{Q}, \vec{p})$$

\vec{q} : état

\vec{Q} : forces appliquées

\vec{p} : paramètres (positions des gouvernes, vent, etc...)

- Solution par intégration numérique

Simulation et temps de calcul sous MATLAB

Lorsque le système $\dot{\vec{q}} = f(\vec{q}, \vec{Q}, \vec{p})$ obtenu est intégré sous MATLAB avec le "slover" `ode45(...)`,

on a quelques problèmes :

- ▶ Les expressions à évaluer son très longues (jusqu'à 20000 caractères)
- ▶ Sur mon Pentium IV, temps relatif de simulation : $\frac{t_{simulation}}{t_{reel}} \simeq 400\%$
- ▶ On est loin du temps réel...



Optimisation et traduction

Programme (MATLAB) d'optimisation :

1) Réduction de la complexité des expressions

- ▶ Recherche de sous-expressions communes ou fréquentes
- ▶ Variables intermédiaires et substitutions
- ▶ Précalcul de fonctions trigonométriques (fréquentes à cause des projections géométriques)

2) Traduction en code C/C++

- ▶ Expressions symboliques MATLAB \rightarrow C/C++ (casse-tête avec chaînes de caractères)
- ▶ Grouper ces expressions dans des fonctions C/C++
- ▶ Génération de fichiers compilables (.cpp et .h)

Intégration en C/C++

L'intégration du système d'équations $\dot{\vec{q}} = f(\vec{q}, \vec{Q}, \vec{p})$, effectuée avec `ode45()` sous MATLAB a également été implémentée en C/C++

Utilisation de la librairie CVODE de la suite SUNDIALS

- ▶ Ordre variable
- ▶ Pas d'intégration variable
- ▶ (Jacobien estimé ou fourni par l'utilisateur)
- ▶ Une version pour processeurs parallèles existe (MACs)
- ▶ ...
- ▶ Tout ce que peut faire `ode45()` et plus

Performances (temps de calcul)

Chaque étape a permis un gain de vitesse conséquent :

- ▶ Intégration et évaluation sous MATLAB

$$\frac{t_{simulation}}{t_{reel}} \approx 400\%$$

- ▶ Intégration sous MATLAB et évaluation en C/C++

$$\frac{t_{simulation}}{t_{reel}} \approx 3\% \text{ à } 16\%$$

- ▶ Intégration et évaluation en C/C++

Encore $\sim 10\times$ plus rapide

Le gain de vitesse final est environ de 1300 et la simulation est largement plus rapide que le temps réel

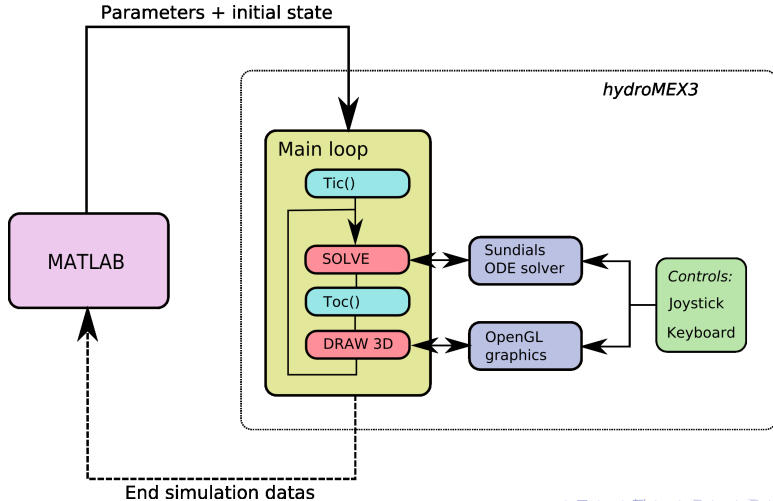
→ Réalité virtuelle possible

Possibilités

Le faible temps relatif de simulation libère les ressources du processeur et nous permet de

- ▶ Réaliser un affichage de la scène en temps réel (OpenGL)
- ▶ Gestion des périphériques (joystick, clavier)
- ▶ (plus tard : implémenter un algorithme de contrôle)

Structure du programme



Indépendance à MATLAB

- ▶ Le programme est un exécutable MATLAB (MEX)
 - ▶ Mais seules les conditions initiales et la récupération des résultats de simulation (éventuelle analyse) sont traités sous MATLAB
- Le programme est indépendant et peut donc facilement être détaché de MATLAB

Documentation

Pour permettre la poursuite du travail

Documentation complète avec Doxygen :

- ▶ Évolutive
- ▶ Version html et pdf
- ▶ Graphes structurels
- ▶ Navigation et hyperliens

Conclusions

- ▶ Objectifs du projet atteints
- ▶ Le programme de traduction est utilisable dans d'autres contextes/projets pour l'accélération de simulations
- ▶ Le simulateur est prêt pour être développé plus avant :
 - ▶ Tables de polaires
 - ▶ Gouvernes au lieu de varier l'angle de calage (Polaires 2D)
 - ▶ Contrôle
 - ▶ ...