Cyrille Piacibello

Développeur

Résidence le Voltaire
Appt 114, Entrée D
234 rue de Suzon, 33400 Talence
06 82 21 11 76
cyrille.piacibello@gmail.com

Expériences

Avril 2017 – Octobre 2021 **Ingénieur d'étude**, *IAP-CNRS*, Paris, Participation au développement du simulateur de l'instrument VIS de la mission spatial EUCLID.
Environnement technique : Python, Linux, git, Jenkins, Redmine, SonarQube

Octobre 2015 – Mars 2017 **Ingénieur d'étude**, *INRIA*, Bordeaux, Implémentation d'un solveur itératif.
Environnement technique : C, C++, OpenMP, Linux, Git

Octobre 2013 – Octobre 2015 **Ingénieur d'étude**, *INRIA*, Bordeaux, Participation au projet ScalFMM.
Environnement technique : C, C++, MPI, OpenMP, Linux, Git

Compétences

Informatique

Langages : C, C++ (norme 2011), Python

Standards : BLAS, LAPACK, MPI, OpenMP

Outils : Shell, LaTeX, Git, Valgrind

Langues

Anglais **lu, écrit, parlé** *Rédaction documentaire. TOEIC 800 en 2012*

Diplômes et Études

2013 **Diplôme Ingénieur**, *ENSEIRB-MATMECA*, Bordeaux.
Spécialisation P.R.C.D : Parallélisme, Régulation et Calcul Distribué

. 2010 **Classes Préparatoires**, *Lycée Massena*, Nice.

Détails des expériences

IAP-CNRS **Ingénieur d'étude**, *Participation au développement du simulateur de l'instrument VIS de la mission spatial EUCLID.*

- Simulation d'effets instrumentaux à partir de modèle et à partir de mesures expérimentales réalisées sur le détecteur VIS.
- Validation des effets implémentés.
- Contribution au Challenge Scientifique. (Passage à l'echelle du wrapper)

Environnement technique: Python, Git, Jenkins, Redmine, Slurm

INRIA **Ingénieur d'étude**, *IB-BGMRes-DR : Implémentation d'un solveur itératif.*

- Implémentation d'un solveur GMRes (multi second membre) par bloc avec produit matrice vecteur externe.
- Détection et prise en compte d'Inexact Breakdown (convergence partielle parmi les seconds membres).
- Deflation au restart (recyclage d'information au restart).

Environnement technique: C++, C, Git

INRIA **Ingénieur d'étude**, *Participation au projet ScalFMM, bibliothèque générique implémentant l'algorithme de Fast Multipole Method.*

- Mise en place d'un ordonnancement par tâche (Standard OpenMP 4.0)

Environnement technique: C++, C, MPI, OpenMP, Git