Projet IF2a C++ : ***Autour du son***

Version 2015

I. Listes de Projets

A. Générateur aléatoire

A partir d'un générateur de nombre pseudo aléatoire, il s'agira de générer des nombres suivant une distribution et des paramètres donnés : Bruit blanc (f_{min} et f_{max}), Gaussienne (moyenne, écart type) et bruit rose (f_{min} et f_{max} et pente). La dynamique des nombres générés sera aussi réglable.

Vérifier la validité de vos générateurs.

Au final, les sorties seront des fichiers wav dont on pourra choisir le format (notamment de représentation des nombres et la fréquence d'échantillonnage).

B. Projet lecture fichier wave + filtrage + écriture wave

A partir des bibliothèques fournies, charger un fichier wave. Programmer un filtrage FIR (ou IIR) dont les coefficients pourront être calculés par Matlab ou grâce aux cours de TS (l'ordre maximal sera 64). Enregistrer le fichier filtré grâce aux bibliothèques fournies.

Il est possible de faire un lecteur pour lire directement le fichier avant et après filtrage.

Il faudra valider votre filtrage sur des gabarits simples.

C. Projet GBF : réalisation d'un générateur de signal

Créer une application permettant de générer des fichiers wave contenant des signaux simples : sinus, cosinus, carré, triangle. Amplitude, offset, fréquence, rapport cyclique doivent être réglables par l'utilisateur et compatible avec l'environnement (format de fichier, matériel). On pourra prévoir une fonction utilisateur (compilée sous forme de librairie et chargée dynamiquement par l'application).

Valider votre générateur de manière numérique et analogique.

D. Projet FFT et DFT : lecture d'un signal et affichage (module et phase)

Calculer et afficher la DFT et FFT d'un signal enregistré dans un fichier wav. Les formats de fichier seront du texte et du wave (pour ces derniers, utiliser les bibliothèques fournies pour les lire). Votre programme permettra à l'utilisateur de choisir le point de départ des transformations et la taille de la fenêtre d'acquisition. On programmera plusieurs formes de fenêtre d'observation.

Valider vos algorithmes et affichages sur des signaux simples ! (sinus modulées, signal rectangle périodique, …).

Etudier les temps de calcul et l'utilisation de la mémoire de ces 2 transformations. Etudier l'impact des fenêtres d'observation.

Les transformations doivent être réversibles.

E. Projet DCT : lecture d'un signal et affichage

Calculer et afficher la Discret Cosine Transform d'un signal enregistré dans un fichier. Les formats de fichier seront du texte et du wave (pour ces derniers, utiliser les bibliothèques fournies pour les lire). Votre programme permettra à l'utilisateur de choisir le point de départ de la DCT et la taille de la fenêtre d'acquisition.

Valider vos algorithmes et affichages sur des signaux simples ! (sinus modulées, signal rectangle périodique, …).

Les transformations doivent être réversibles.

Attention : il faudra pouvoir transformer tout le signal contenu dans le fichier : si la fenêtre est plus petite que le signal, il faudra appliquer plusieurs fois de suite la transformation en se décalant dans le signal.

F. Compression dynamique et Gate d'un signal wav

La compression dynamique permet d'ajuster la dynamique d'un signal dynamiquement… c'est-à-dire en fonction de l'intensité courante du signal. Il s'agira de créer un programme permettant d'effectuer une compression dynamique sur un fichier wav. Les paramètres de la compression (seuil, ratio, temps d'attaque et de retour) seront réglables et illustrés dans l'application. Il sera de même avec le Gate. Pour les deux traitements, il faudra aussi choisir la manière de mesurer « l'intensité courante du signal » (utiliser qu'une seule valeur est très sensible aux bruits).

Valider vos traitements sur des signaux adaptés ! (signal rectangle périodique, bruit …)

G. Réverbération acoustique d'un signal wav

Le but est d'appliquer à un fichier wav une réverbération numérique par convolution. L'utilisateur doit pouvoir choisir les différents paramètres et la réponse impulsionnelle (ou autre mesure de la fonction de transfert).

Valider vos algorithmes sur des signaux simples ! (sinus modulées, signal rectangle périodique, …)

H. Compression de fichier wav par SVD et FFT

Faire un programme permettant de compresser un fichier wav et de lire un fichier compressé. Pour la compression, votre programme met en forme le signal (matrice) puis applique la SVD. La mise en forme se fera soit directement à partir du signal, soit par FFT de fenêtres successives (valeurs complexes, organisées ligne à ligne en suivant la taille de la fenêtre utilisée pour la FFT).

Vous vous appuierez sur la GNU Scientific Library (GSL) pour réaliser ces traitements.

Le but sera de proposer un algorithme complet de compression et décompression. Pour chaque approche, vous illustrerez l'impact du nombre de composantes utilisées pour la représentation du signal (et le taux de compression) sur la qualité du signal compressé.

II. Consignes communes de programmation

A. Outils

- Langage de programmation : C++, C++11
- Langage de modélisation : UML (2.0, diagramme de classes, au besoin : diagramme d'objets)
- IDE de programmation : QtCreator
- Librairies de calculs : STL, sauf si précisions spécifiques mentionnées dans le sujet
- Librairie d'optimisation : *OpenMP*, *thread* du C++
- Librairies IHM graphique : Qt (version 4 ou 5)
- Classes interfaces utilisateurs et graphiques séparées des classes algorithmes.

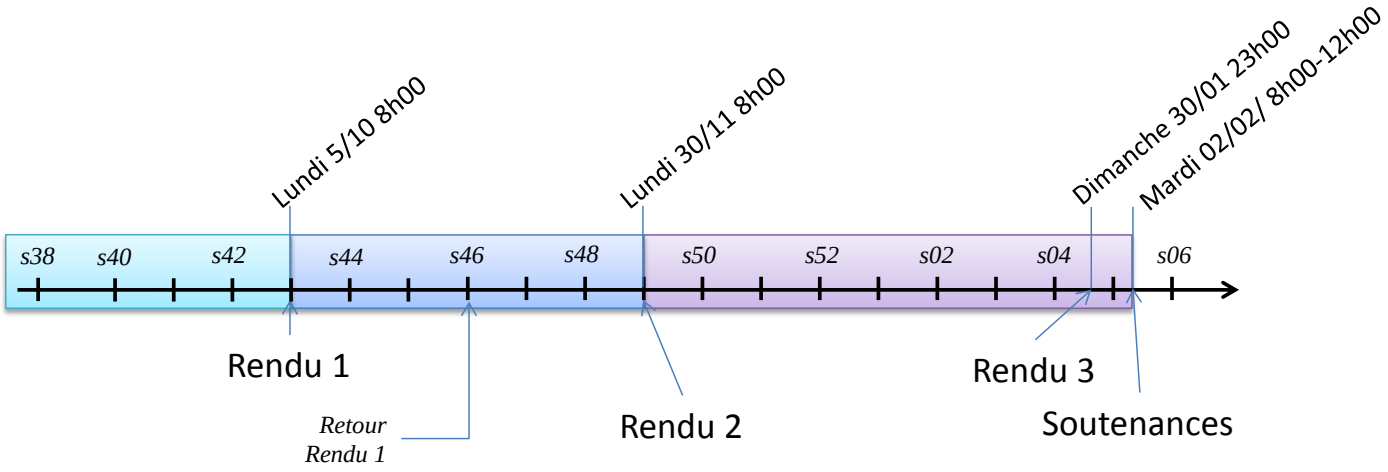
B. Recommandations et hygiène de codage

- Faire des classes dédiées au calcul et des classes d'adaptation pour l'IHM,
- Commenter votre code, pour les fonctions et clase utilisez le style Doxygen (www.doxygen.org)
- Utiliser une indentation rigoureuse,
- Les noms de variables et des fonctions doivent être cohérents entre UML et C++, et bien choisis…
- Pas de variables globales, aucun goto
- Attention aux fuites mémoires (utiliser le débogage et *valgrind* avec qtcreator)

III. Consignes de conduite du projet

Les projets sont à faire par groupe de 3 à 4 personnes. Les rendus sont communs au groupe et se font sur moodle. L'ensemble de l'équipe doit valider les rendus sous moodle.

Attention : de nombreux rendus sont attendus à des dates précises et fixes. Ils se font tous sous *moodle*. Voici les différents jalons et les dates des différents rendus :



A. Phase 1 et Rendu 1 : organisation du travail

Ce rendu sera corrigé 2 à 3 semaines après le dépôt. En cas de non validation de ce rendu, il sera à resoumettre. La seconde phase de travail ne pourra être validée tant que ce rendu n'est pas validé. Seule la version finale de ce rendu sera prise en compte dans l'évaluation.

1. Création de l'équipe et choix du projet.

Sur moodle2, il faudra choisir votre équipe. Une équipe est constituée de 4 personnes et est directement liée à un projet (correspondance des lettres). La répartition est au choix des étudiants et est définitive.

2. Cahier des charges (2 pages max)

Le premier travail est d'établir un cahier des charges et les spécifications importantes de votre projet. A valider par l'enseignant.

3. Cahier de recette (2 pages max)

Avec le cahier des charges, donnera les objectifs de réalisation et les tests à conduire pour valider le fonctionnement de votre application. Pour ce projet il s'agira principalement de l'évaluation quantitative des résultats produits par votre application. A valider par l'enseignant.

4. Organisation du travail (2 pages max)

Diagramme de Gantt, répartitions des tâches de chaque membre du groupe, système de travail collaboratif retenu, méthode de validation interne des contributions.

B. Phase 2 et Rendu 2 : étude et conception

Ce rendu sera retourné annoté aux étudiants. En cas de non validation par l'enseignant ce rendu sera à resoumettre. La troisième phase de travail devra s'appuyer et être conforme aux propositions définies dans ce rendu. Seule la version finale de ce rendu sera prise en compte dans l'évaluation.

1. Diagramme de classes (1 à 2 pages)

La modélisation de votre application sous forme de diagramme de classes uniquement. Au besoin d'autres diagrammes pourront être ajoutés (cas très particulier). Penser à justifier vos choix et respecter les consignes relatives à la programmation.
Le diagramme de classe est à faire valider.

2. Solutions à mettre en œuvre (4 pages max)

Il s'agit de donner les éléments techniques, algorithmiques que vous avez retenus et que vous allez mettre en œuvre. Sont attendus les éléments clés de votre projet concernant les explications mathématiques, illustrations algorithmiques, détails d'optimisation, détails de réalisation IHM… Ne pas tout présenter : focalisez-vous sur l'important.

C. Phase 3 et Rendu 3 : Réalisations et Rendu final

A rendre sur moodle au plus tard 3 jours avant la soutenance…

Le rendu final est composé:

- des sources
- du rapport
- des transparents de la soutenance

1. Les sources

Une archive zip contenant les fichiers sources et le fichier de projet (pas les exécutables) doit être fournie et posée sous *moodle*.

Votre projet sera recompilé et exécuté: les données d'exemple et tous les fichiers sources doivent être présents dans cette archive. Vérifiez que votre projet se compile bien sur d'autres machines (type celles de GE).

L'archive des sources est à déposer sur *moodle*.

2. Le rapport

Document en français ou en anglais, format pdf, rédigé, illustré et mis en page (numérotation des parties, des pages, alignement du texte justifier, légende aux figures et tables, sommaire, page de garde, …).

Il ne doit pas comporter le code. Seuls de courts extraits sont peut être utiles au cours de votre discours ou en annexes. Vous pourrez réutiliser les rendus 1 et 2.

Le but du rapport est d'expliquer le projet et son déroulement, de détailler les développements réalisés (principalement UML, quelques extraits de C++ vraiment si nécessaire), les tests effectués pour garantir une « bonne » fiabilité de l'application ainsi que l'adéquation aux objectifs initiaux du projet.

La structure du rapport pourra être la suivante :

- Introduction : présentation du sujet, objectifs du projet et structure du rapport
- Contexte : cahier des charges, contexte mathématiques, physiques, algorithmique, …
- Réalisation : diagramme UML, solution algorithmique, IHM, gestion du projet (répartition du travail, organisation, outils de travail collaboratif), réalisations annexes (validation)
- Expérimentations : définitions des tests, des mesures de validation, résultats des tests et discussions des résultats
- Conclusion (qui en plus de faire un bilan sur la réalisation, pourra justifier des écarts entre les rendus 1 et 2 et le rendu 3)

3. La soutenance

Tout comme le rapport, le but de la présentation est d'expliquer, à l'ensemble de l'auditoire, le projet et son déroulement, les développements réalisés, la fiabilité de l'application et l'adéquation par rapport aux objectifs. L'auditoire doit être convaincu (et donc avoir les éléments nécessaires) que la réalisation est fiable et efficace. (Imaginer une présentation devant votre direction ou des financiers, en vue d'un financement ou d'une promotion ou reconduite de poste).

La présentation du projet au reste de la promotion se fera en 10 minutes. Suivront 10 à 15 minutes de questions. Les questions porteront sur tous les points de la réalisation et de la conduite du projet (répartition du travail) et pourront être adressées explicitement à un membre du groupe.

Une démo peut être faite dans le cadre de la présentation (dans les 10 minutes).

Les supports de cette présentation (10 transparents) seront déposés sur moodle au plus tard la veille de la soutenance.

Chaque étudiant doit assister aux présentations des autres groupes et évaluera objectivement et avec les mêmes critères que le jury d'enseignants, les réalisations des autres groupes.

Il est possible d'inviter votre tuteur entreprise aux soutenances.

IV. Critères d'évaluations du Rendu 3

Chaque critère est évalué par un score de satisfaction : 0, 1, 2, 3 ou 4. '4' correspondant à une satisfaction totale.

A. Les sources et réalisation

- Hygiène de codage (indentation, noms des variables-fonctions-classes, commentaires, gestion mémoire, …)
- facilité de compilation,
- exécution et fiabilité,
- qualité de l'IHM
- cohérence entre conception et réalisation
- réalisme par rapport aux objectifs du projet

B. Le rapport

- La forme, le style, la clarté, les illustrations, l'orthographe/grammaire/conjugaison,
- Les solutions d'implémentation retenues (pertinences, adéquations, efficacités),
- Les résultats et leurs analyses (validation).

C. La soutenance

- Respect du temps de présentation,
- Qualité des supports de présentation (clarté, efficacité, lisibilité, adaptés),
- Pédagogie : explication du contexte, des solutions choisies et implémentés, les résultats obtenus et la discussion sur ces résultats,
- Confiance dans la qualité de la réalisation (lors de la présentation, avez-vous eu les éléments suffisants pour juger de la qualité du travail ?),
- Acceptabilité de la réalisation par rapport au cahier des charges,
- Qualité de la gestion de projet,
- Qualité des réponses aux questions.

La note de projet est individualisée. Elle prendra en compte les 3 rendus, une évaluation individuelle ainsi que l'évaluation des étudiants par les étudiants et l'implication dans le projet.

L'évaluation individuelle de projet sera faite dans le cadre de l'examen écrit d'IF2.