2eme Rapport d’avancement

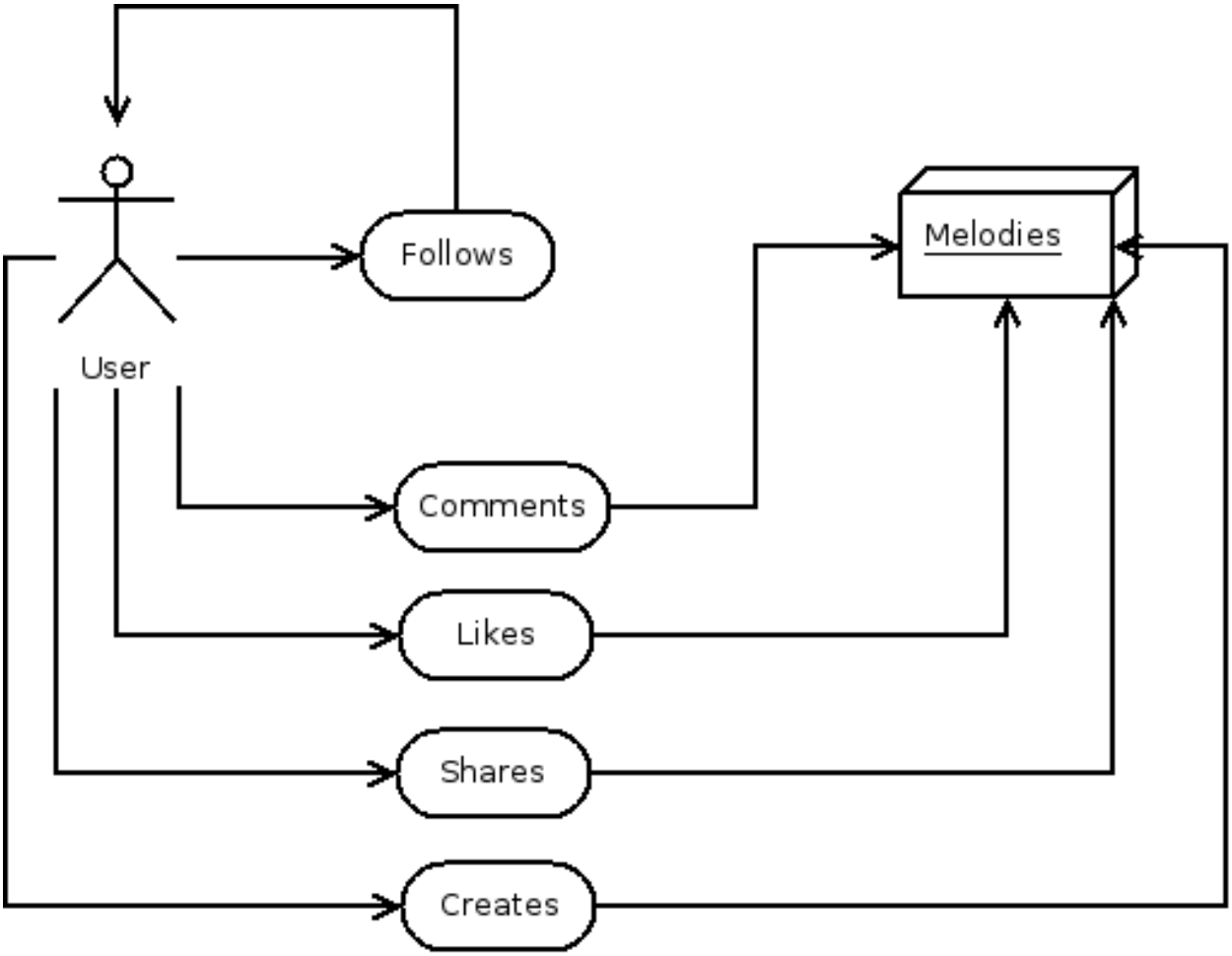
Projet de Fin d’études : Sujet 2

Réalisé par :
Zaher Hamadeh
Nicolas Jbeyli

Encadré par :
Rima Kilany Chamoun

I - Fonctionnalités de l'utilisateur.

L'application doit permettre à l'utilisateur d'effectuer un certain nombre d'actions. Voici les fonctionnalités que notre application offre.

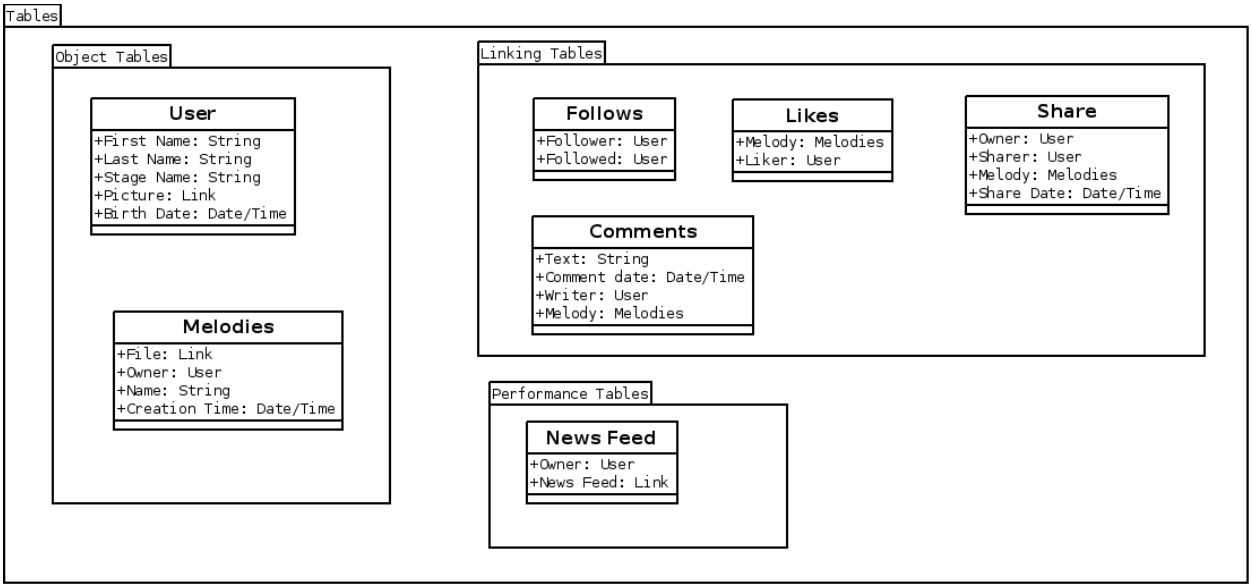


A ce stade, l'utilisateur peut crée des melodies et suivre un autre utilisateur. Le développement de ces fonctionnalités se fait en parallèle avec l'avancement du développement du code coté serveur. Les trios fonctionnalités qui restent a implementer sont similaires et ne devraient pas nécessité un temps énorme.

L'application offre à l'utilisateur une vue pour créer la mélodie, une vue pour suivre un autre utilisateur et une vue pour commenter, aimer et partager une mélodie. Il nous reste alors une unique vue a developper.

II - Structure de la base de données.

La base de données doit refléter les actions que peut effectuer l'utilisateur dans l'application. Voici la structure adoptée :



Les tables Objets représentent les objets que nous manipulons. Les tables de lien sont présenté pour sauvegarder les actions commises par les utilisateurs. Les table de Performance sont la pour améliorer cette dernière. La table news feed contient un lien vers un fichier qui contient toutes les actions commises par les utilisateurs que suit notre client. Elle permet une rapidité de la réponse et une bonne experience pour l'utilisateur.

III- Reconnaissance

III-1 Clustering

Cette méthode est une amélioration de la troisième. Elle consiste à diminuer la taille de la fenêtre, pour obtenir en moyenne 40 à 50 fenêtres (8 dans la méthode précédente). Pour une fenêtre i, on récupère la fréquence dont la puissance est maximale, ce qui créera un couple (F_i, P_i). Les couples (F_i, P_i) forment l'ensemble F. On calcule l'ensemble FF tel que:

$$FF = \{(F_d, P_d) \mid F_d = F_j - F_i \quad P_d = P_j \times P_i \quad ((F_i, P_i), (F_j, P_j) \in F ; i < j)\}$$

Etablir FF nécessite un algorithme de complexité $O(n^2)$, $n \approx 40$ dans notre cas on peut admettre.

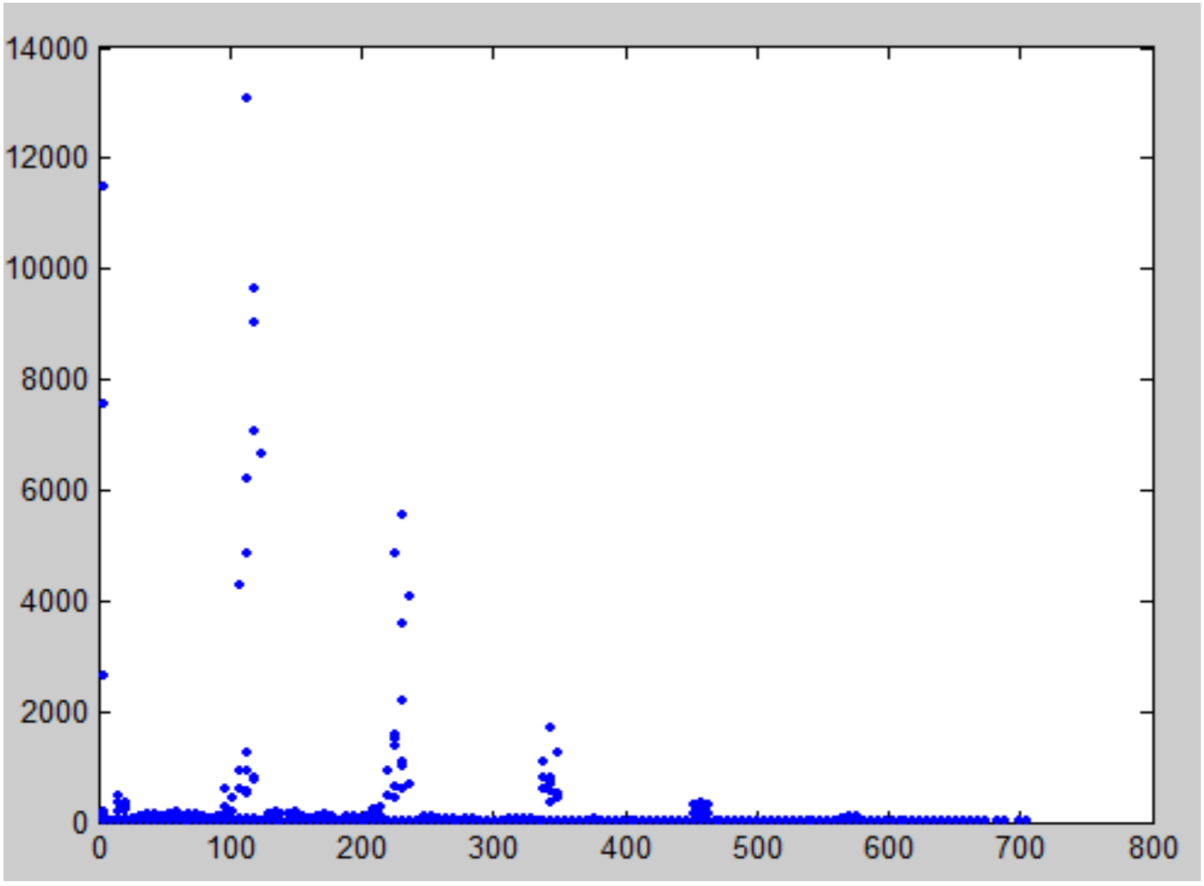


Fig 4. Graphe représentant l'ensemble FF. F_d en abscisse, P_d en ordonnée.

On peut éliminer certaines valeurs au-dessous d'un seuil. On obtient ainsi quelques points qui peuvent être regroupés en cluster.

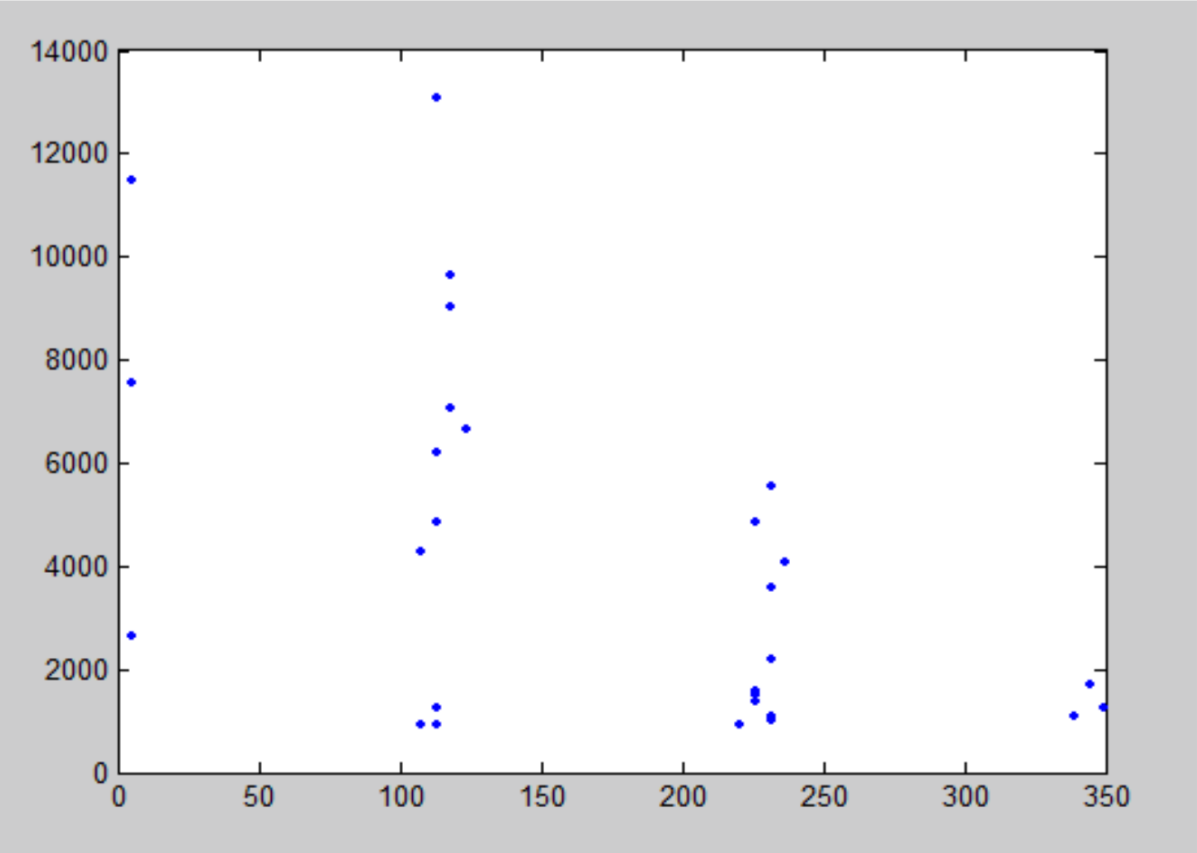


Fig 5. Graphe de FF, avec les points dont la puissance est inférieure à un seuil sont supprimés.

Les points sont donc clairement rassemblés en cluster. Grâce à une méthode inspirée du K- means cluster, on calcule le centre de chaque cluster. Les points proches du zéro sont négligés. La complexité d'un algorithme de K-means clustering est $O(n^{dk+1}\log(n))$ avec d le nombre de dimensions (2) et k le nombre de cluster (ne peut être fixe). Ce calcule est lourd et non- nécessaire.
les caractéristique spécifique d'un signal musicale nous aident à établir un algorithme heuristique inspiré du k-mean clustering. Le centre de fréquence minimale est donc la fréquence fondamentale.

IV- Raffinement

IV-1 Contexte

Jusqu'à l'instant, nous avons réussit à extraire d'un fichier audio les fréquences formant les notes chantées avec un résultat assez satisfaisant. Pourtant, un chanteur débutant ne trouvera pas notre application amusante puisque, vu qu'il chante faux, il obtiendra des résultats scientifiquement justes mais non plaisants.
Dans cette partie, nous allons plonger dans les méthodes de raffinement et d'amélioration d'une mélodie, pour qu'elle soit musicalement acceptable.

IV-2 Récupération des limites des notes

Un chanteur débutant peut commencer à chanter une note, puis glisser et la changer sans faire attention tout en croyant qu'il chante toujours une même note.

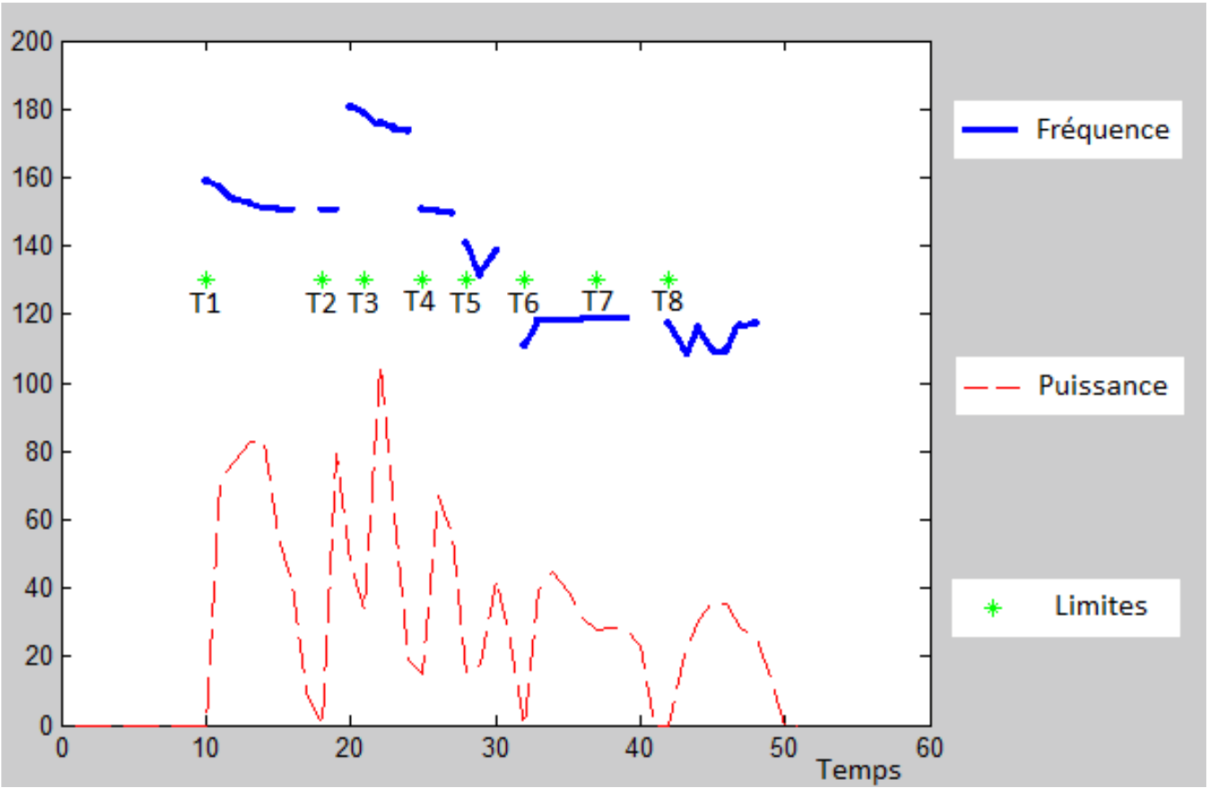


Fig 6. Graphe d'une mélodie (phrase de la chanson « 7 nation Army »)

D’après la figure 6, une seule note est supposée être chantée entre T1 et T2. Cependant, la courbe bleue nous indique que le chanteur est entrain de glisser du au contrôle médiocre de sa voix. La courbe rouge nous indique la puissance du signal au cours du temps. Grâce à la dérivée de la puissance, on récupère les limites des notes chantées. Ainsi, la note chantée entre T1 et T2 est la moyenne des fréquences récupérées dans cet intervalle.

IV-3 Auto tune

L’auto tune est un outil utilisé par un grand nombre de chanteur populaire de nos jours. Il consiste à corriger les notes chantées, en les attribuant à la note la plus proche appartenant à une gamme définit.
Par exemple, la gamme de Do majeur contient les notes Do, Ré, Mi, Fa, Sol, La, Si. Les notes (Do dièse, Mi bémol...) sont considérer comme étranger à la gamme. Tandis que la gamme de Ré majeur contient les notes Ré, Mi, Fa dièse, Sol, La, Si, Do dièse. Un chanteur manquant de l’expérience chantera des notes étrangères. Ces notes seront corrigées pour appartenir aux gammes probables.