Date création : 13/12/2015 11:24:00

Nao RAL – documentation technique

I. NAO : l'interface utilisateur

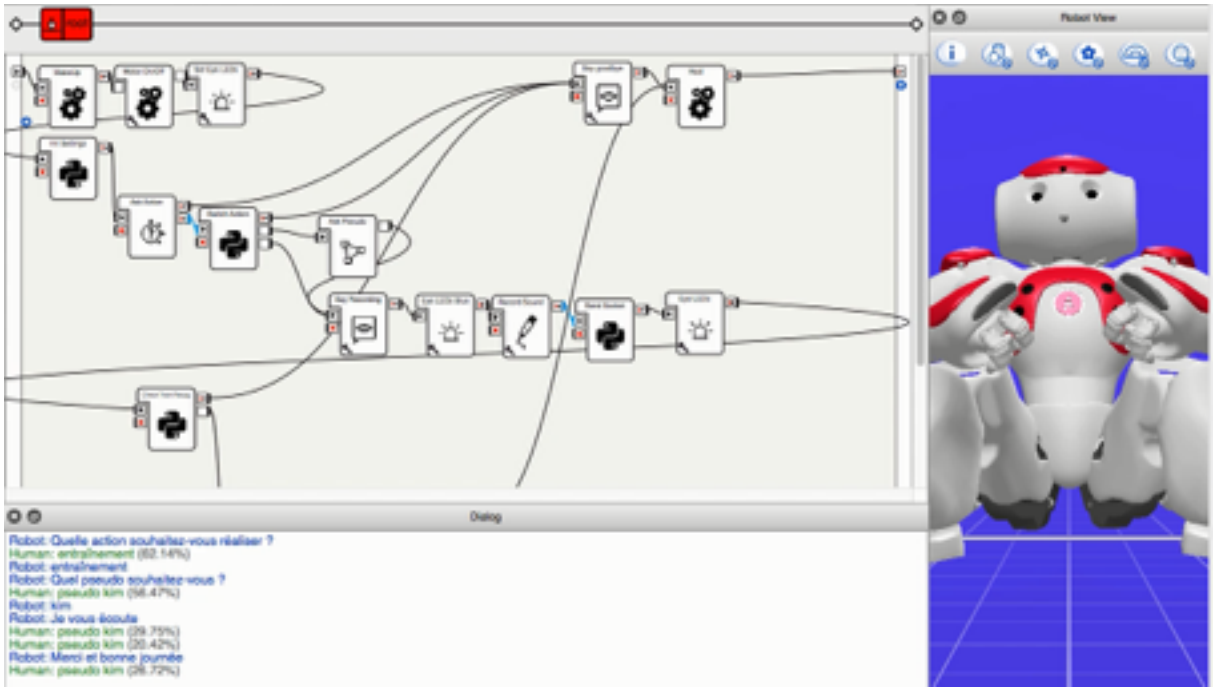
Dans le bloc « **Init settings** » :

- Renseigner l'adresse IP de NAO ainsi que son port d'écoute pour les sockets TCP/IP
- Indiquer l'adresse IP du serveur ainsi que son port d'écoute pour les sockets TCP/IP

Dans le bloc « **Ask Pseudo** » :

- Ajouter le pseudo de l'utilisateur s'il n'est pas disponible dans le dictionnaire

A. Entraînement



Lancer l'application sur Choregraphe :

- Dire « entraînement » : action = « train »
- Dire « pseudo <pseudo> » : user_pseudo = « <pseudo> »
- Prononcer une phrase (5 secondes) = échantillon de la voix

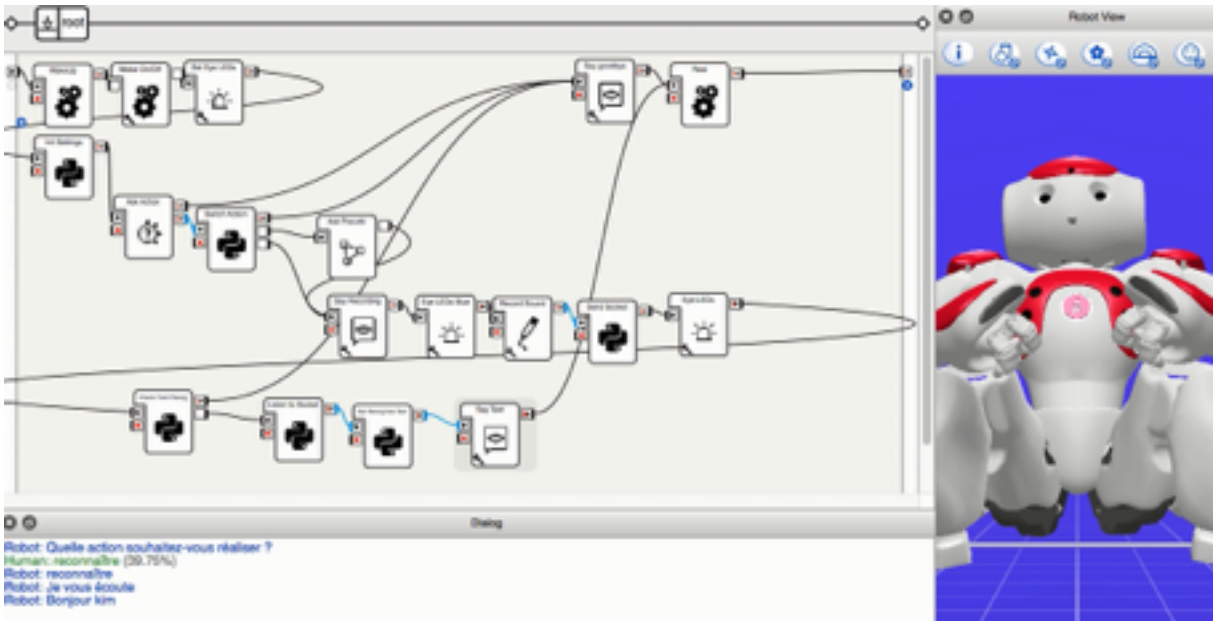
Nao enregistre l'échantillon sous le chemin suivant : recordings/microphones/ train_<user_pseudo>_<timestamp>.ogg

Nao crée la socket TCP/IP : <nao_ip>|<nao_port>|train_<user_pseudo>_<timestamp>.ogg

```
169.254.232.44|40000|train_kim_145346231961.ogg
```

➔ Nao envoie la socket TCP/IP au serveur

B. Reconnaissance



Lancer l'application sur Choregraphe :

- Dire « reconnaître » : action = « recognize »
- Prononcer la phrase (5 secondes) = échantillon de la voix

Nao enregistre l'échantillon sous le chemin suivant :
recordings/microphones/train_anonymous_<timestamp>.ogg

Nao crée la socket TCP/IP : <nao_ip>|<nao_port>|train_anonymous_<timestamp>.ogg

➔ Nao envoie la socket TCP/IP au serveur

II. Serveur Python : le TCP/IP socket listener

```
('169.254.232.44', 35581) connected
169.254.232.44|40000|train_kim_145346231961.ogg
NAO host 169.254.232.44
NAO port 40000
SOCKET : data 0 = 169.254.232.44
SOCKET : data 1 = 40000
SOCKET : data 2 = train_kim_145346231961.ogg
Close

Process finished with exit code 0
```

Serveur Pyhton qui a l'unique rôle d'écouter les sockets TCP/IP sur le port 30 000.
S'il reçoit bien la socket avec le format <ip>|<port>|<nom_fichier> alors il récupère chaque champ délimité par « | ».
On a alors dans l'ordre des champs :

- 0 : l'adresse IP de Nao
- 1 : le port d'écoute de Nao pour les sockets TCP/IP
- 2 : le nom du l'échantillon

Chaque champ est stocké dans la matrice **files_to_download.mat** de la façon suivante :

169.254.232.44	40000	train_kim_145346231961.ogg

III. Serveur Matlab : l'intelligence artificielle

Tous les paramètres de configuration sont indiqués dans **ral_settings_init.m**
Les paramètres n'ont pas été choisis au hasard.
En effet, une batterie de tests a été effectuée afin d'ajuster ces différents paramètres et obtenir les meilleurs résultats possibles. Vous trouverez ci-dessous les résultats obtenus :

	Arnaud_exist	Arnaud_test	Florian_exist	Florian_test	Kim_exist	Kim_test	Maylis_exist	Maylis_test	Edwin-intrus
coupe_sil = 0,04	0.98661 0.97321 0.97768	0.89573 0.90521 0.89573	0.86017 0.86864 0.89831	0.72321 0.63095 0.64881	0.93467 0.92965 0.90452	0.93168 0.85714 0.85714	0.93304 0.95536 0.97321	0.78595 0.77582 0.7893	0.62044 0.52555 0.5438
coupe_sil = 007	0.98925 0.99462 0.98925	0.94253 0.91379 0.94828	0.97256 0.96682 0.97256	0.74476 0.74825 0.76923	0.90698 0.96512 0.89535	0.84884 0.84884 0.89535	0.93589 0.96273 0.95031	0.75424 0.83051 0.80508	0.59004 0.54406 0.63985
Fech = 8000	0.97321 0.94196 0.98214	0.87356 0.85057 0.86782	0.93467 0.94975 0.95467	0.62178 0.66189 0.56734	0.86869 0.83838 0.9697	0.43243 0.52703 0.54054	0.96482 0.94472 0.90955	0.79518 0.79518 0.73092	0.28916 0.41365 0.39357
Neur cachés = 50	1 0.98206 1	0.90521 0.92961 0.90947	0.91964 0.95089 0.95536	0.74405 0.71726 0.73619	0.92547 0.96894 0.99379	0.86765 0.86029 0.90441	0.97487 0.9799 0.9598	0.78182 0.80657 0.78667	0.5977 0.59387 0.55619
FIR 0,97 - Cs 0,07	1 0.99462 0.99462	0.93103 0.94828 0.93103	0.97256 0.97256 0.99052	0.75175 0.75874 0.6958	0.90698 0.94186 0.94186	0.83838 0.85859 0.83838	0.95652 0.91304 0.95031	0.80932 0.76695 0.77119	0.5364 0.59387 0.5977
nb neur = 50 (ss train)	0.99462 0.99462 0.99462	0.99462 0.99462 0.99462	0.98578 0.98578 0.98578	0.77273 0.77273 0.77273	0.89535 0.89535 0.89535	0.83828 0.83828 0.83828	0.96273 0.96273 0.96273	0.83475 0.83475 0.83475	0.54789 0.54789 0.54789
nb neur = 100	1	0.92954	0.99052	0.8007	0.94186	0.90909	0.95789	0.77119	0.47126

A. Daemon

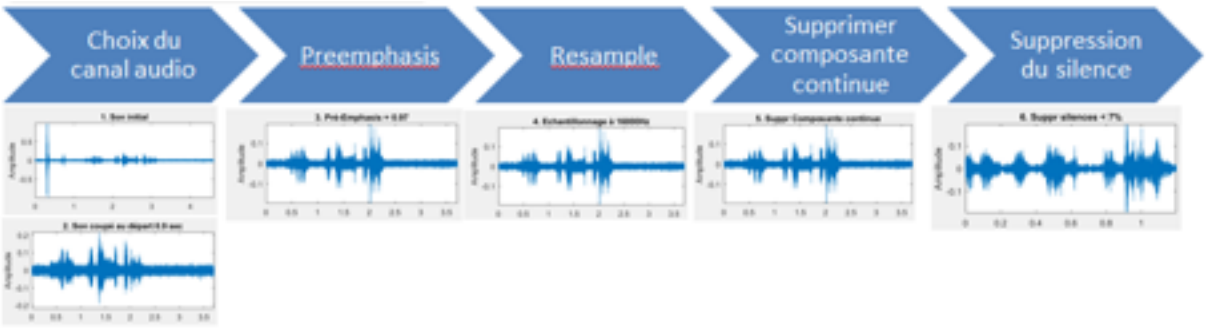
Récupère les lignes de la matrice **files_to_download.mat**
Télécharge, par FTP, les échantillons indiqués. Ils sont extraits du dossier recordings/ microphones/ de NAO.
Stocke les échantillons téléchargés dans le dossier **audi_inputs/**

B. Création d'une signature vocale

Cepstre : transformation d'un signal du domaine temporel vers un autre domaine analogue au vers un autre domaine temporel.
MFCCs : Mel-frequency cepstral coefficients, les vecteurs acoustiques caractérisant une voix, la signature vocale d'une personne.

Caractères physiques de l'interlocuteur : taille larynx, forme de la bouche, puissance des sons…

1. Préparation du signal avant l'extraction des MFCC



Choix du canal audio :

- Si 4 canaux, alors on Nao comme micro : on prend le 3ème canal
- Si 2 canaux, alors on est en stéréo : on prend le 1er canal

Preemphasis : pré-accentuation du signal

Resample : ré-échantillonage à 16 000 Hz si nécessaire

Supprimer composante continue : recentrer le son sur l'axe des abscisses

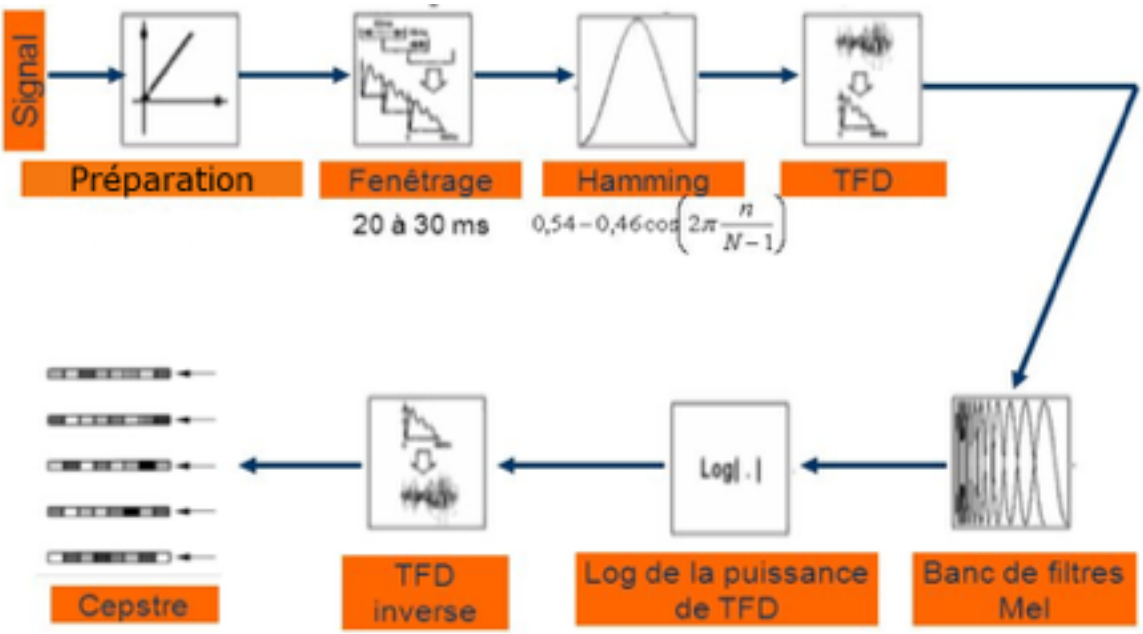
Suppression du silence :

- Découpage de l'échantillon par frame de 1 seconde
- Repère la valeur mac : la valeur de l'amplitude la plus grande
- Pour chaque frame, si l'amplitude du signal ne dépasse pas strictement 7% (silence_threshold) de cette valeur max alors on supprime cette frame de l'enregistrement

Nous n'avons pas besoin du silence puisque notre étude porte sur les coefficients de la voix de l'interlocuteur.

➔ Nous obtenons donc un signal épuré

2. L'extraction des MFCC



Préparation : étapes décrites ci-dessus

Fenêtrage : découpage de l'échantillon en frame de 30 ms

Hamming : prétraitement avant d'appliquer la transformation de Fourier
Pour observer un signal sur une durée finie, on le multiplie par une fonction fenêtre d'observation => quand on multiplie un signal $s(t)$ par cette fenêtre, on n'obtient plus que la partie comprise entre T_1 et T_2 de l'échantillon

TFD = Transformation de Fourier Discrète : convertir chaque frame qui sont dans le domaine du temps en frame dans le domaine de fréquence
Discret = signal non continu
Calcul sur des échantillons, signal digitalisé
☒ on obtient un spectre du signal

Banc de filtres Mel : comme l'Homme ne perçoit pas les sons de façon linéaire. L'espace entre un son plus ou moins aigu n'est pas tout le temps le même à l'oreil.
L'échelle des Mel permet de traduire cette perception : HZ <-> Mel
En fonction de N Mel, on perçoit que le son est plus ou moins aigu
☒ conversion du signal avec l'échelle de Mel

Log de la puissance TFD et TFD inverse : extraction des MFCCs pour chaque frame

➔ ensemble des MFCCs = vecteur accoustique
Stocke les MFCCs dans **ral_db_mfcc.mat**
Stocke les utilisateurs dans **ral_db_users.mat**

C. Réseau de neurones

Comme on a stocké les utilisateurs avec leurs MFCCs associés, on peut entraîner un réseau de neuronnes : le **ral_net**.
Ce dernier est constitué d'une couche intermédiaire à 50 neuronnes.
Son but est d'atteindre un pourcentage d'erreur inférieur ou égal à 0.000000001
S'il n'y arrive pas, alors il s'arrêtera après 1000 phases d'entraînement.

Une fois la phase d'entraînement finie, il est possible de demander au réseau de neurones d'identifier une voix
Pour chaque utilisateur enregistré, on obtient un taux de ressemblance.
La somme des taux est égale à 1, voici un exemple :

Toto	Tata	Lorem	Ipsum	Kim
0.1	0.1	0.05	0.047	0.703

Si un taux dépasse 70% alors on peut affirmer que la voix appartient à telle utilisateur.
Nous avons donc reconnu la personne. Nous récupérons son pseudo.
Si aucun taux ne dépasse la limite, alors nous déduisons qu'un intrus a essayé d'utiliser notre système. On utilisera le pseudo « inconnu » dans ce cas-là.

Une fois que la personne est reconnue, on envoie son pseudo à Nao via une socket TCP/IP.
Nao dira alors « Bonjour <pseudo> ».