Documentation du logiciel P300 Stimulator

Guillaume Gibert

10/12/2008

# Structure du répertoire

Il y a deux répertoires : \_beta et \_fcs qui contiennent une version du stimulateur P300. Dans le répertoire \_beta, il s’agit d’une version en cours de développement a priori instable. Dans le répertoire \_fcs se trouve la version courante du stimulateur.

Dans chacun de ces répertoires il y a 7 sous-répertoires.

Les sous-répertoires et leurs fonctions sont :

* Bin : contient l’exécutable nommé P300Stimulator
* Doc : contient la documentation du logiciel
* Include : contient tous les fichiers <\*.h> nécessaires à la compilation du logiciel (remarque : il y a un sous répertoire dans Include nommé vrpn qui contient tous les fichiers <\*.h> de la fonctionnalité vrpn)
* Lib : contient tous les bibliothèques et les classes compilées (fichiers <\*.o>) nécessaires au bon fonctionnement du logiciel (remarque : il y a 3 bibliothèques vrpn <\*.a> dans ce dossier)
* Share : contient les polices de caractères utilisées par le logiciel (sous-répertoires font), l’icône du logiciel (sous-répertoire img), la liste des mots donnée à une personne (sous-répertoire list) et les scenarios utilisés (sous-répertoire scenario)
* Src : contient tous les fichiers <\*.cpp> nécessaires à la compilation du logiciel
* Test : contient le script de démarrage start.csh

# Structure du logiciel

Le logiciel est composé de 10 fichiers <\*.cpp> correspondant chacun à une classe différente.

En détails, nous avons :

* ParallelPort qui dérive de la classe Port, elle permet de gérer le port parallèle.
* VGAPort qui dérive de la classe Port, elle permet de gérer le port VGA.
* XmlParser qui permet de parser le fichier scenario au format xml.
* VrpnListener qui permet d’implémenter un client vrpn
* Tools qui permet de créer des séries de valeurs aléatoires et de déterminer la fréquence du CPU.
* Timer qui permet d’implémenter différents types de chronomètres plus ou moins précis.
* FreeType qui permet la création de police spécifique de type FreeType dans un environnement SDL.
* Display qui permet la création d’une fenêtre SDL avec les bonnes caractéristiques.
* P300Stimulator qui gère le scénario et flashe la matrice de lettres comme décidé dans le scenario.

Bien sûr, chacun des fichiers <\*.cpp> a son fichier include <\*.h> qui se trouve dans le répertoire include.

# Structure des fichiers scenario

Les fichiers scenario sont au format xml. Voilà un exemple de scenario :

<experiment>

<settings>

<mode>online</mode>

<fullScreen>no</fullScreen>

<wordPrediction>no</wordPrediction>

<vrpn>openvibe-vrpn@localhost</vrpn>

<trigger>no</trigger>

<display>

<screen>

<width>1024</width>

<height>768</height>

<bpp>16</bpp>

</screen>

<backgroundColor>

<red>0.0f</red>

<green>0.0f</green>

<blue>0.0f</blue>

</backgroundColor>

<items>

<number>36</number>

<kind>letter</kind>

<font>ArenaBlack.ttf</font>

<shape>matrix</shape>

<row>6</row>

<column>6</column>

<vspace>100</vspace>

<hspace>130</hspace>

<size>30</size>

<color>

<red>0.5f</red>

<green>0.5f</green>

<blue>0.5f</blue>

</color>

<list>ABCDEFGHIJKLMNOPQRSTUVWXYZ123456789\_</list>

</items>

<flash>

<enhanced>0</enhanced>

<color>

<red>1.0f</red>

<green>1.0f</green>

<blue>1.0f</blue>

</color>

<numberByItem>1</numberByItem>

<order>random</order>

<duration>100</duration>

<interval>200</interval>

<pauseIntraItem>5000</pauseIntraItem>

</flash>

</display>

</settings>

<run>

<pause duration="2000">La stimulation va commencer</pause>

<target>95</target>

<pause duration="5000"> </pause>

</run>

</experiment>

Le fichier doit commencer par la balise <experiment> et se terminer par la balise </experiment>.

Ensuite, il y a deux grandes sections : la section « settings » et la section « run ». La section « settings » correspond aux paramétrage du stimulateur alors que la section « run » correspond à la séquence de pause et mots à traiter.

Dans la partie « settings », on doit choisir :

* Mode : online ou offline (si online, on va demander au serveur vrpn de fournir une répons eaprès chaque série de flashs)
* Fullscreen : yes ou non (si oui, la matrice prendra tout l’écran)
* wordPrediction : yes ou non (pour l’instant option non développée par défaut la valeur est non)
* vrpn : <nom du périphérique>@<machine où tourne le serveur vrpn>
* trigger : yes ou non (si oui, envoie de trigger sur le port parallèle, attention il faut être root sous linux pour avoir cette option)
* display :
  + screen :
    - width : largeur de la fenêtre
    - height : hauteur de la fenêtre
    - bpp : nombre de bits par pixel
  + backgroundColor
    - red : quantité de rouge dans la couleur de fond
    - green : quantité de vert dans la couleur de fond
    - blue : quantité de bleu dans la couleur de fond
  + items
    - number : nombre d’items dans la matrice
    - kind : pour le moment seule l’option lettre est implémentée
    - font : police choisie (vérifier qu’elle existe dans le répertoire share/font)
    - shape : seule l’option matrix est active pour le moment
    - row : nombre de lignes dans la matrice
    - column : nombre de colonnes dans la matrice
    - vspace : espace en pixels entre les lignes
    - hspace : espace en pixels entre les colonnes
    - size : taille des lettres
    - color
      * red : quantité de rouge dans les lettres au repos
      * green : quantité de vert dans les lettres au repos
      * blue : quantité de bleu dans les lettres au repos
    - list : liste des items à afficher dans la matrice
  + flashs
    - enhanced : grossissement des lettres lorsqu’elles sont flashées
    - color
      * red : quantité de rouge dans les lettres flashées
      * green : quantité de vert dans les lettres flashées
      * blue : quantité de bleu dans les lettres flashées
    - numberByItem : nombre de flash par ligne ou colonne
    - order : ordre des flashs, seul l’ordre random est implémenté
    - duration : durée des flashs en ms
    - interval : ISI en ms
    - pauseIntraItem : pause entre 2 lettres flashées

Dans la partie « run », on doit choisir :

* Soit une pause avec la balise <pause>, on doit rajouter la durée de cette pause par exemple <pause duration=  « 2000 »>, la pause durera 2s. Si on veut qu’un texte s’affiche lors de la pause on le rajoutera entre les balises <pause> et </pause>
* Soit un mot à épeler entre les balises <target> et </target>

# Compilation

Il faut aller dans le répertoire bin, puis taper

$ make clean

$ make

# Utilisation

## Locale (jouer un scenario)

Il faut se positionner dans le répertoire bin et taper :

$./P300Stimulator ../share/scenario/<nom du scenario.xml>

## Globale (jouer une liste de scenarii)

Il faut se positionner dans le répertoire test et taper :

$./start.csh ../share/list/<nom de la liste>