Travaux Pratiques de Traitement Vidéo :
Manipulation d'un *Blue Screen*

6 septembre 2005

Résumé

Ce TP a pour but de vous initier aux techniques de traitement vidéo. L'objectif est de réaliser un *plug-in* de manipulation de *blue screen*, capable de modifier l'image d'arrière plan d'une vidéo en fonction de son niveau sonore. Une description bas niveau du contenu de la vidéo (XML) sera également produite pour les TPs suivants (moteur de recherche vidéo).

Pour ce faire, nous utiliserons la plate-forme *libre* Transcode. Votre *plug-in* est une bibliothèque liée dynamiquement au coeur de Transcode et sera appelé à chaque traitement de trame. Il prendra donc en entrée soit une trame vidéo (une image au format RGB24 à modifier), soit une trame audio (séquence PCM de 40 ms).

FIG. 1 – Vidéo avant et après traitement

1 Pour commencer

- Copiez l'archive du TP `/usrsrv/share/iim_video/tp_video.tar.gz` vers le répertoire `/tmp` ;
- Décompressez l'archive (`$ tar xvzf tp_video.tar.gz`) ;
- Déplacez vous jusque dans le répertoire `tp_video/stuff` ;
- Pour compiler et installer votre *plug-in*, entrez la commande suivante : `$ ./install` (depuis le répertoire `tp_video/stuff`) ;
- Pour tester votre *plug-in*, entrez la commande suivante : `$ ./test` ;
- Pour visualiser le résultat : `$ xine test.avi` ;
- Le code source à éditer (`filter_iim.c`) se trouve dans le répertoire `tp_video/stuff/transcode-1.0.0/filter`. N'oubliez pas de sauvegarder ce fichier dans votre répertoire personnel à l'issue du TP ;
- La vidéo à traiter se trouve dans le répertoire `tp_video/data/videos/cvideo2.avi`. Un exemple de résultat final peut être trouvé dans `tp_video/data/videos/final.avi`.

2 Prise en main

2.1 Bonjour tout le monde !

Votre *plug-in* traite la vidéo image par image. Il dispose d'une interface unique :

```
int tc_filter(vframe_list_t *ptr, char *options)
```

où `ptr` pointe une structure de données complexe représentant une trame vidéo. Une description peut être trouvée dans le fichier :

```
tp_video/stuff/transcode-1.0.0/src/framebuffer.h
```

Votre filtre peut être appelé dans divers contextes, dont :
- `TC_FILTER_INIT` : au chargement du *plug-in* ;
- `TC_FILTER_CLOSE` : au déchargement du *plug-in* ;
- `TC_VIDEO` : pour traiter une image ;
- `TC_AUDIO` : pour traiter une trame audio.

Exercice 1 *Modifiez le* plug-in *de sorte à afficher "*Bonjour tout le monde*" à chaque fois que Transcode l'invoque pour traiter une image.*

2.2 Lecture/écriture du buffer

La structure `vframe_list_t` possède un champ (`video_buf`) qui matérialise l'image à traiter. Les pixels y sont représentés de la manière suivante :
- Chaque pixel est codé sur 3 octets consécutifs (mode RGB24) ;

- Les pixels sont codés les uns à la suite des autres ;
- L'image est mémorisée ligne par ligne ;
- Les lignes sont codées les unes après les autres ;
- Le tout forme un tableau de caractères à une dimension, de longueur :

```
ptr->v_width * ptr->v_height * 3.
```

FIG. 2 – Représentation mémoire d'une image RGB24

Exercice 2 *Modifiez le code de manière à affecter la couleur noire à tous les pixels de la vidéo. Faites de même avec la couleur grise, puis blanche.*

Exercice 3 *Modifiez le code pour afficher les trois composantes de chaque pixel. Que pouvez-vous en conclure sur l'encodage des composantes ?*

3 Manipulation du *blue screen*

Exercice 4 *Écrivez une fonction permettant de déterminer si un pixel est "bleu" ou non (attention à l'encodage des composantes).*

Exercice 5 *Écrivez une fonction qui remplace tous les pixels bleus par des pixels rouges.*

Votre *plug-in* dispose de fonctionnalités permettant de charger des images (format PNM) en mémoire (fonction `load_pnm_picure(char *path)`). De plus, vous disposez d'un tableau de chaînes de caractères référençant les chemins vers les images disponibles pour ce TP (variable `picture_list`).

Exercice 6 *Modifiez le code pour que votre* plug-in *charge l'une des images dans le buffer* `background_buf` *lors de son initialisation.* ***Attention****, n'oubliez pas de libérer le buffer lors du déchargement du* plug-in.

Exercice 7 *Écrivez une fonction qui remplace chaque pixel bleu par le pixel correspondant dans l'image chargée dans le buffer* `background_buf`.

4 Analyse de la piste audio et interaction avec le *blue screen*

Dans le cadre de ce TP, la piste audio est encodée en PCM stéréo 16 bits à 44,1 kHz.

Le format PCM est un format de représentation brut des échantillons sonores. Le champ `audio_buf` de la structure `aframe_list_t` matérialise une suite d'échantillons, durant 40 ms.

Chaque échantillon est donc codé sur deux octets. Le pointeur `a_sample` vous permettra de parcourir facilement ce buffer. La taille de ce dernier est donnée (en octets) par le champ `audio_size` de la structure `aframe_list_t` (cette structure est désignée par le pointeur `aptr`).

Le buffer est structuré de la manière suivante : les échantillons sont stockés les uns à la suite des autres, le canal gauche entrelacé avec le canal droit (G,D ;G,D ;G,D ;G,D ;...).

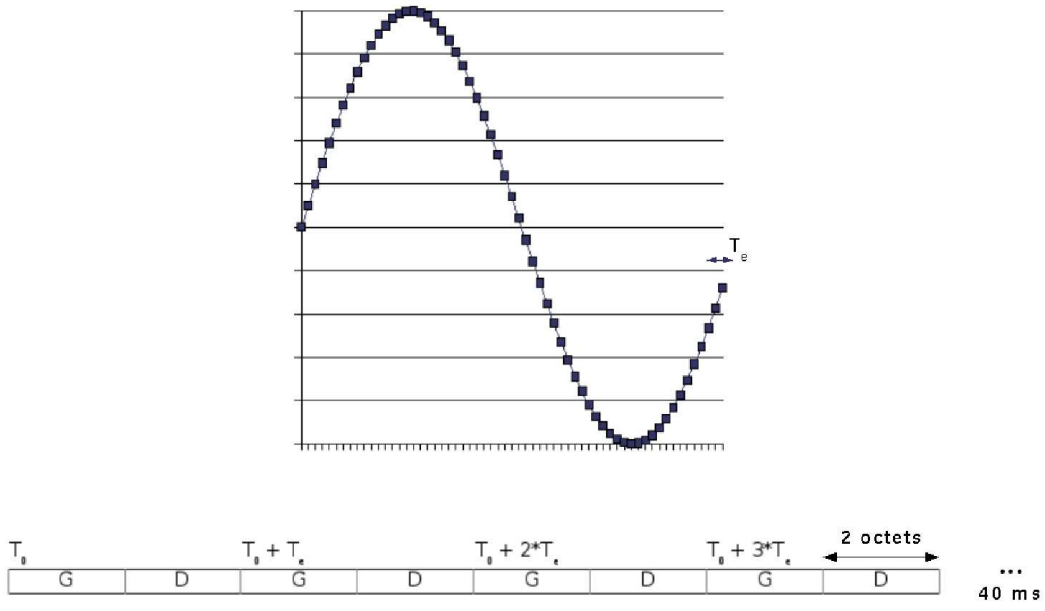

FIG. 3 – Représentation mémoire d'une trame audio

Exercice 8 *Écrivez une fonction permettant de calculer le carré de l'amplitude de chaque échantillon.*

Exercice 9 *Écrivez une fonction permettant de calculer la moyenne sur une trame du carré de l'amplitude de chaque échantillon.* ***Attention****, l'utilisation d'une variable déclarée en* `long long int` *risque d'être nécessaire.*

Votre *plug-in* n'est chargé qu'une fois en mémoire. La fonction `tc_filter` est en revanche régulièrement invoquée par Transcode. Le mot clé `static` permet d'allouer une zone mémoire jusqu'à la terminaison du programme, sans en modifier le contenu.

Ainsi, pour stocker un résultat intermédiaire entre deux trames, utilisez des variables déclarées en `static`.

Exercice 10 *Écrivez une fonction permettant de détecter les fortes variations positives du volume sonore d'une trame à l'autre.*

Exercice 11 *Écrivez une fonction permettant de changer l'image de fond lorsqu'un important bruit de fond* commence *sur la vidéo (voir `tp_video/data/videos/final.avi`).*

5 Génération d'une description de contenu

L'analyse bas-niveau de la vidéo que vous avez réalisée (c'est-à-dire le découpage en segments selon le niveau sonore) peut être utile à des fins de description. De telles descriptions (suivant le standard MPEG-7) sont notamment utilisées dans les moteurs de recherche vidéos.

Il vous est demandé dans cette partie du TP de générer un document XML récapitulant les attributs bas niveau de chaque segment, à savoir :

- L'identifiant de la trame de début de séquence ;
- La durée de la séquence (en nombre de trames) ;
- Le nom du fichier utilisé en image de fond.

La structure `video_segment_desc_t` a été mise à disposition pour mémoriser ces informations. La variable `video_desc` est un tableau de descriptions de segment, de taille suffisante.

La fonction `save_description` vous permettra de sauvegarder automatiquement votre description au format MPEG-7 une fois la variable `video_desc` remplie. Cette fonction prend également en paramètre le nom de la vidéo de sortie. Celui-ci est donné par la variable `vob->video_out_file`.

Exercice 12 *Modifier le code du* plug-in *de sorte à mémoriser les attributs de chaque segment. **Attention**, une fois la description de la vidéo complétée, pensez à initialiser à `NULL` le pointeur `background` de la case suivant la dernière case renseignée, pour marquer la fin du tableau (cf. fonction `save_description`).*

Le fichier XML obtenu comporte, pour chaque segment, un champ `FreeTextAnnotation`. À vous d'y entrer quelques mots clés décrivant la portion de vidéo concernée (cf. TPs suivants).

6 Bonus

Pour ceux qui se sont ennuyés au cours de ce TP...

Exercice 13 *Écrivez une fonction permettant de modifier les composantes d'un pixel connaissant ses coordonnées cartésiennes.*

Exercice 14 *Écrivez une fonction qui permet d'afficher le logo ENIC en haut à droite de la vidéo (`tp_video/data/pictures/logo.pnm`).*

Exercice 15 *Écrivez une fonction qui permet de n'afficher que les lignes inférieures (selon un paramètre) du logo ENIC en haut à droite de la vidéo.*

Exercice 16 *Modifiez le* plug-in *de sorte que le logo ENIC apparaisse progressivement en haut à droite au début de la vidéo et disparaisse en fin de vidéo (cf. `tp_video/data/videos/final.avi`).*

Exercice 17 *Modifiez le* plug-in *de sorte que l'animation avec le logo apparaisse en semi-transparence (cf. `tp_video/data/videos/final.avi`).*