Université de Cergy-Pontoise

Projet Mouvements de Patterns
Cahier des Charges

David Barouh
Marc Teyssier
David Liebgott

Tuteurs :
Mr Philippe Laroque
Mr Pierre Andry

Année 2007-2008
Projet de Master 2
V0.98

Table des matières

1 Introduction

Ce projet a été proposé par les membres de l'équipe DMD qui sont David Barouh, Marc Teyssier et David Liebgott. Les tuteurs messieurs Philippe Laroque et Pierre Andry nous ont aidés à proposer un sujet réalisable par rapport à nos connaissances et aux temps dont nous disposons.
Notre projet est la détection de mouvements de patterns.
L'équipe sera en autogestion, l'ensemble de ses membres devra veiller au bon fonctionnement du projet.
Ce document décrit les choix fonctionnels et techniques que nous avons fait. Il a été réalisé en fonction du temps imparti que nous avons pour la réalisation du projet. Toutes les fonctionnalités additionnelles seront dans la section cahier des charges avancés. Si le planning nous le permet nous les implémenterons autant que possible dans notre projet.

2 Cahier des charges minimal

Le but du projet est de piloter des applications externes appelées Binap (Binary Is Not A Plugin) grâce aux mouvements des mains.
Pour ce faire, un flux vidéo sera envoyé à un serveur qui traitera les images et renverra un code en fonction des mouvements réalisés par l'utilisateur. Les applications peuvent être diverses, piloter un navigateur Web, des servo-moteurs pour faire bouger une caméra ou encore piloter une télévision.

FIG. 1 – Schéma du principe fonctionnel

3 Cahier des charges fonctionnel

Le projet devra au moins répondre aux fonctionnalités contenues dans cette partie.

3.1 Capture vidéo

L'utilisateur ce placera devant une Webcam qui capturera la vidéo. Celle-ci sera branchée sur un ordinateur. Elle communiquera avec le serveur de traitement via le protocole TCP/IP (en Ethernet ou en Wifi). Il n'y a ici aucun traitement de la vidéo, elle sera uniquement envoyée sur le réseau.

3.2 Le serveur

Le serveur recevra la vidéo et devra la traiter pour communiquer les actions faites par l'utilisateur aux applications externes. Toutes la partie calcul sera embarqué sur le serveur.

3.3 Les applications pilotables

3.3.1 Administrateur

L'Interface Homme-Machine (IHM) sera une application externe de type administrateur. Elle permettra de faciliter le dialogue entre la "machine" et l'utilisateur.

Elle sera composée d'un "joystick" pour la main droite afin d'avoir au minimum les 8 directions :

- haut droite
- haut
- haut gauche
- gauche
- bas gauche
- bas
- bas droite
- droite

Une fois la sélection faite, le joystick se recentrera automatiquement.
On pourra avec la main gauche choisir l'application externe que l'on souhaite piloter. Pour ce faire, on utilisera un cercle que l'on placera sur l'icône correspondante à l'application. Une fois l'application sélectionnée, il reviendra à sa position d'origine et l'icône sélectionné sera marqué. Ensuite il suffira de commander l'application avec le joystick.
Cette Interface Homme-Machine sera considérée comme une application administrateur, elle utilisera le protocole détaillé plus loin.
Cette application étant très spécifique elle sera nécessairement sur la même machine (que ce soit ordinateur ou non) que la Webcam.
Un seul administrateur pourra ce connecter sur le serveur.

3.3.2 Externes

Dans un premier temps les applications externes ne pourront se connecter au serveur qu'à l'initialisation. Dans les versions supérieures, elles devront pouvoir se connecter dynamiquement à n'importe quel moment. Les applications décoderont le message passé puis le convertiront en action en fonction de leur application.
Les applications pourront être de plusieurs sortes et piloter n'importe quel appareil. Elles seront complètement décorrélées du serveur et du traitement de l'image, elles recevront uniquement un message d'information du mouvement ou de la commande à effectuée.

4 Choix technologiques

4.1 Le flux vidéo

Le flux vidéo entre le système de capture et le serveur sera transporté via le protocole TCP/IP qui permet de garantir une connexion uniquement avec le serveur, contrairement à l'UDP. Le message sera transporté avec une liaison Ethernet filaire ou Wifi. Le choix a été fait d'envoyer la vidéo image par image, ou au moins dans un flux totalement créé par nos soins. En effet, ceci nous permettra de compresser à la volée la vidéo suivant la qualité du réseau.

4.2 Serveur

On utilisera le langage C++ pour coder l'ensemble du traitement de l'image et les émissions/réceptions sur le réseau.
Le flux vidéo du réseau sera lu puis l'image sera traitée afin de placer des points de suivi (tracking) sur les éléments de sélection gauche et droit. Un ensemble de procédure sera fait sur ces éléments afin de les nettoyer des erreurs et de pouvoir les afficher correctement. Pour nous aider dans l'utilisation des points de tracking et de l'image en générale, nous utiliserons la librairie open source OpenCV optimisée par Intel.
Une fois la position des éléments décryptée un message sera envoyé à l'application externe sélectionnée via le réseau. Nous avons créé un protocole propre à notre application.

4.3 Protocole

Nous avons développé notre protocole dans le but de dialoguer avec l'Interface Homme-Machine (ou une application "administateur") mais aussi avec les applications externes.

Pour avoir les détails de ce protocole, voir le document annexe "Protocole".

4.4 Interface et applications externes

4.4.1 IHM

Elle sera basée sur la librairie OpenCV qui permet, grâce à un ensemble de fonctionnalités, d'afficher simplement et rapidement les images et d'ajouter des éléments. L'IHM étant nécessairement sur le même matériel que la capture vidéo, cela nous évite de lui envoyer le flux vidéo et permet ainsi de réduire le débit utilisé sur le réseau. De plus, l'utilisateur se plaçant devant la webcam, il est donc logique que l'affichage se fasse sur le même matériel.
L'IHM embarquera par défaut un jeu d'icônes type qui pourront être affichées.

4.4.2 Externes

Les applications externes pourront être programmées en n'importe quel langage capable de se connecter à une adresse IP par un socket.
Pour fonctionner, ces applications devront bien évidemment implémenter notre protocole.

5 Les risques

Le projet demande principalement des connaissances en traitement d'image, en réseau et en langage de programmation (notamment le C++). C'est pourquoi l'équipe sera très vigilante sur ces différents points en essayant d'avoir le plus grand nombre de solutions possibles pour parer aux différents problèmes. De plus, la gestion de projet et la gestion du planning permet d'anticiper au maximum ces risques.

5.1 Traitement de l'image

Aucun des membres de l'équipe n'a de solide base de traitement d'image. C'est pourquoi nous avons décidé d'utiliser une librairie qui nous offre des fonctionnalités de base. De plus, nous utiliserons le moins possible de techniques de traitement d'image, ceci afin d'éviter d'avoir des problèmes sur des fonctionnements que nous ne maîtrisons pas.
Le risque dans notre application est que les éléments de sélections ne réagissent pas correctement à cause de l'image capturée par la webcam. Pour parer aux problèmes de reconnaissances, il faudra que l'utilisateur se place dans un environnement le plus neutre possible, c'est-à-dire sans différences important de contraste dans l'image (pas de lumière visible dans l'image, les objets foncés sur un fond claire ...). Il est conseillé aussi d'éviter la lumière néon qui fait des "scintillement" sur la capture de l'image.

Toutefois, nous n'excluons pas de mener de nouvelles études sur le traitement d'image en fin de projet, suivant son avancement.

5.2 Programmation en langage C++

Le traitement de l'image utilise des processus lourd, c'est pourquoi nous avons choisi le langage C++ qui nous permet de faire du langage objet et qui possède des meilleurs performances que le Java. De plus il a une communauté active de développeur où nous pouvons trouver des aides face aux difficultés que nous rencontrerons.

6 Cahier des charges avancé

Dans cette partie nous détaillerons toutes les idées d'évolutions que nous avons eu. Ces fonctions ne sont pas nécessaire pour que le système fonctionne correctement.

6.1 Gestion dynamique des applications externes

Ajouter la possibilité que les différentes applications puissent se connecter au serveur à n'importe quel instant. Il faudra que celle-ci s'affiche sur l'interface d'administration avec l'icône correspondant. De plus, les applications pourront se déconnecter sans empêcher la continuité du fonctionnement global du système, et le serveur devra agir en conséquence pour modifier l'interface et les données envoyées sur le réseau.

6.2 Utilisation d'un PDA

Le système de capture d'image et l'IHM sont embarqués sur un matériel de type PDA, ceci afin d'augmenter la portabilité de l'application. Il en résulte des contraintes de performances (un PDA n'ayant pas les mêmes ressources qu'un ordinateur de bureau). De plus, il faudra compiler le projet pour le PDA.

6.3 Type de sélection avec le joystick

Pour augmenter les possibilités d'utilisation du joystick en ajoutant la possibilité de rester sur une direction sans qu'il se recentre. L'utilisateur aura deux niveaux de sélection : Le premier où les informations seront envoyées en continue et une pour n'en envoyer qu'une seule.

6.4 Utilisation sans affichage

Si le système ne détecte pas l'application d'administration IHM, le pilotage se fera avec une reconnaissance des formes pour la main qui choisi l'application externe et un mouvement de main droite pour la piloter. Exemple : pour piloter la télévision, il faut avoir le poing fermé. Ensuite un geste vers le haut de la main droite qui aura pour effet d'augmenter le son. Pour une seconde application il faudra avoir la main à plat pour la piloter etc.

6.5 Système de simulation de souris

On peut imaginer que la main ouverte est une souris évoluant sur un ecran. Pour cliquer, il pourra suffir de fermer le poing. On pourrait ainsi complètement se passer des éléments de sélection, et pourquoi pas remplacer un système de pointage complet.

6.6 Apprentissage du mouvement

Le système pourrait être capable d'apprendre une suite de mouvement que l'on pourra relier à une action d'un Binap. Ceci permettrait de minimiser le nombre d'action à faire.

6.7 Gestion d'un grand nombre de BINAP

Si le nombre d'application externe est trop grand (supérieure à trois), l'affichage pourra être modifié pour ajouter une navigation des applications connectées grâce à des flèches haut et bas.

6.8 Application externe

On peut ajouter au système autant d'application externe que voulu pour peu qu'elles respectent le protocole. Ensuite libre à elle d'effectuer les actions en conséquence.

6.9 Détection des doigts

Dans une version évoluée, on pourra détecter la position exacte des doigts afin d'offrir un nombre de possibilité de sélection plus important. On pourrait notamment traduire le langage des signes.

7 Documentation

7.1 Documents de gestion de projet

- Cahier des charges
- Plan assurance qualité

7.2 Documents livrables

- Manuel utilisateur
- Manuel technique
- Site Internet