Cahier des charges : Tableau virtuel interactif

Baptiste Saleil Geoffrey Mélia Julien Pagès Kevin Bollini

15 février 2012

Table des matières

1 Introduction **3**
- 1.1 Fonctionnement général . . . 3
- 1.2 Finalités . . . 3
- 1.3 Choix stratégiques . . . 3
  - 1.3.1 Librairie - Application . . . 3
  - 1.3.2 Différentes étapes . . . 3

2 Contexte **4**
- 2.1 Faisabilité . . . 4
- 2.2 Existant . . . 4
- 2.3 Ressources . . . 4

3 Analyse et conception **4**
- 3.1 Organisation . . . 4
- 3.2 Planning . . . 4
- 3.3 UML . . . 4

4 Librairie **5**
- 4.1 Fonctionnalités . . . 5
- 4.2 Architecture . . . 5
- 4.3 Outils . . . 5
- 4.4 Étapes . . . 5

5 Application **5**
- 5.1 Fonctionnalités . . . 5
- 5.2 Architecture . . . 6
- 5.3 Outils . . . 6
- 5.4 Étapes . . . 6

6 Conclusion **6**

1 Introduction

1.1 Fonctionnement général

Le but de l'application est de simuler une écriture ou un dessin sur un tableau virtuel. Ceci en se basant sur une reconnaissance de mouvements via une webcam.

1.2 Finalités

Pour ce projet universitaire, l'objectif est d'obtenir un produit qui soit réellement utilisable. La finalité est donc d'avoir la possibilité de pouvoir utiliser rapidement l'application avec un étalonnage simple et rapide.
De plus l'intérêt est également d'avoir une application utile qui peut par exemple être utilisée dans le cadre d'un cours, pour rapidement dessiner un schéma et l'exporter et le sauvegarder ensuite.

1.3 Choix stratégiques

Pour mener à bien ce projet, nous avons choisi de le découper. Ce découpage nous permettra de mieux nous organiser, de mieux répartir les tâches, et donc de fournir un travail plus efficace, même sur le long terme.

1.3.1 Librairie - Application

Notre premier choix consiste à découper notre projet en deux projets bien distincts, une librairie de suivi de mouvements, et une application graphique.

Ce choix représente plusieurs intérêts :

- Premièrement, ce découpage permet de bien différencier les tâches, le développement de la librairie requiert des connaissances en traitement d'images pour tracer des couleurs/objets, appliquer des filtres, manipuler les images, etc... ce qui correspond parfaitement à la formation de deux membres du groupe. (Formation IMAGINA Université Montpellier 2). Le développement de l'application, lui, requiert des connaissances en génie logiciel, pour l'architecture de l'application, en IHM, pour la réalisation de l'interface (logicielle et gestuelle), ou encore en réseau pour la mise en place de l'architecture client/serveur. Ces différents points correspondent également parfaitement à la formation des deux autres membres du groupe. (Formation AIGLE Université Montpellier 2). Ainsi, les tâches peuvent bien se répartir en fonction des connaissances et spécialités de chacun.

- Deuxièmement, l'intérêt de ce découpage est d'avoir deux projets complètement indépendants. En effet, la librairie sera développée d'un côté et permettra d'offrir un bon nombre de fonctionnalités de suivi ou manipulation d'image. L'application, elle, se chargera d'utiliser les fonctionnalités de traitement d'image de la librairie en ajoutant une interface, une couche réseau, une connexion aux webcam, la récupération du flux, etc... La librairie et l'application sont donc deux projets développés en parrallèles et très modulaires, notre application peut très bien utiliser une autre librairie de traitement d'image, et la librairie peut très bien être utilisée par d'autres application qui offriraient des fonctionnalités totalement différentes de la notre.

1.3.2 Différentes étapes

Notre deuxième choix à été de découper le projet en plusieurs étapes. Arrivés en Master 1, nous avons maintenant quelques projets à nos actifs et savons maintenant comment éviter certaines erreurs. L'objectif final du projet est assez haut, ce choix d'étapes nous permet donc d'avoir un premier objectif accessible, puis d'ajouter des fonctionnalités à la librairie, et à l'application petit à petit pour franchir les étapes et s'approcher au maximum de nos objectifs. On pourrait par exemple se fixer les étapes qui suivent :

- Prototype fonctionnel avec côté librairie une reconnaisance basique de forme ou mouvement, et côté application une interface basique permettant d'accéder à la webcam, d'appeler la librairie, de dessiner sans personnalisation, ainsi qu'une ébauche du module réseau. Ce prototype permettra de poser les bases de l'architecture de l'application et de la librairie.
- Ajout d'algorithmes de reconnaissance et de nouvelles méthodes de reconnaissance pour la librairie, et amélioration du module réseau (définition du protocole, application serveur fonctionnelle, gestion des paquets, etc...)

- Optimisation des algorithmes de trace et optimisation des méthodes. Ajout de l'interface gestuelle et personnalisation du dessin.
- Finalisation du code, préparation d'une utilisation en production. Compilation dynamique de la librairie, création de paquets linux (.deb,...) compilation sur plusieurs systèmes d'exploitation.

2 Contexte

2.1 Faisabilité

Choix d'un projet en fonction de nos spécialités, de notre formation, expérience, ce qu'on sait faire.

2.2 Existant

- Exemples dans le jeux vidéo : Kinect, Eye-toy, CamSpace. (Techniques pour l'IHM, par exemeple, mouvements continus ou immobilité sur une zone)
- Exemples sur les thèses et projets de Recherche (Techniques de programmations)

2.3 Ressources

Documents de supports (thèses, articles etc...)

3 Analyse et conception

3.1 Organisation

Le projet se déroulera autour de trois dates clés :
- 2 Mars 2012 : Bilan de mi-parcours.
- 27 Avril 2012 : Bilan de pré-soutenance.
- 14 Mai 2012 : Rendu complet du projet.

Nous devons donc fixer nos objectifs en fonction de celles-ci, en essayer de réaliser la première etape, le prototype fonctionnel, autour de la première date.

Détail des étapes, répartition du travail, outils de synchronisations/communications ...

3.2 Planning

Rétro planning gantt commenté.

3.3 UML

Analyse avec diagramme de classe, use case, séquence, etat ?

4 Librairie

4.1 Fonctionnalités

Le but de cette librairie serait de fournir des fonctions permettant le repérage et le suivi d'un objet en particulier à partir d'images. L'idée serait donc d'avoir dans la librairie un maximum de fonctions de traçage différentes, de la plus lente et précise à la plus rapide et permissive, de manière à adapter la complexité en fonction du degrés de précision requis pour notre traitement.
On peut alors imaginer deux classes de fonctions, des la plus basiques réalisant un tracking intuitif, sans mémoriser d'information sur l'objet suivi, jusqu'à des fonctions plus perfectionnés, effectuant un repérage préliminaire pour connaitre l'objet. Le repérage devra permettre à partir d'une image et d'une position polaire de reconnaître et d'isoler un objet. Cette étape permettra de récuperer un certain nombre de caractéristiques sur l'objet qui permettront ensuite de le suivre, comme par exemple sa couleur, sa taille etc.
Dans un premier temps, nous implémenterons une fonction basique de suivi par couleur, retournant la position polaire de l'objet sur l'image.
D'autre part, nous essayerons d'inclure a la librairie des fonctions de détections d'actions permettant par exemple de signaler que l'on écrit par exemple.

4.2 Architecture

Architecture de la librairie
- Fonctions simples : On trace une forme ou une couleur, sans apprentissage, à partir de deux images et d'une position.
- Fonctions "intelligente" : On commence par recueillir des informations (forme, taille, ancienne localisation ou couleur par exemple) sur l'objet tracé que l'on conserve ensuite
- Fonctions de detections d'actions (Mouvement/transformation signifiant une action par exemple main qui se ferme pour clic)

L'objectif final serait de fournir cette librairie sous forme de bibliothèque dynamique d'abord au format UNIX .so et éventuellement en .dll pour windows.

4.3 Outils

Nous utiliserons le langage C/C++ et la librarie OpenCV pour le développement de la librairie, qui sera elle-même destinée à C. Notre choix s'est porté sur ce langage pour différentes raisons :
- Nous souhaitons utiliser la librairie OpenCV qui à été initialement créée pour le langage C.
- L'application sera elle aussi faite en C++.
- C'est le langage que nous connaissons et affectionnons le plus.

Nous avons décidé d'utiliser OpenCV car c'est une librairie puissante, très bien documentée et fortement orientée sur la vision par ordinateur.

4.4 Étapes

Détail des étapes avec fonctionnalités pour chacune d'elle.

5 Application

5.1 Fonctionnalités

L'application doit permettre en premier lieu de faire rapidement un schéma ou écrire quelques phrases et de le sauvegarder : ce qui n'est pas possible avec un tableau conventionnel.
Dans l'optique d'obtenir une application réellement fonctionnelle, permettra de dessiner et d'écrire sur le tableau virtuel avec une interface Homme-Machine naturelle.

- L'exportation du dessin ou de l'écriture vers un format pdf, ou image
- Une gomme permettant d'effacer
- Un module réseau pour faire un travail collaboratif avec l'application
- Plusieurs formes de pinceaux avec plusieurs couleurs

5.2 Architecture

Architecture de l'application

5.3 Outils

Nous utiliserons des librairies pour accélérer le développement de l'application, et donc d'obtenir plus rapidement un résultat abouti. L'application contient une interface graphique, il nous faut donc une librairie pour cela : notre choix s'est porté sur Qt, pour plusieurs raisons :

- La librairie est très puissante et complète, avec une bonne documentation
- Elle est multi-plateformes
- C'est une librairie que nous connaissons déjà un peu, cela réduit donc le temps passer à se former dessus
- C'est une librairie essentiellement C++, ce qui s'adapte au langage choisi pour l'application, et pour la librairie

Nous utiliserons également QtDesigner pour nous assister dans la création de l'interface graphique (placement des éléments...).

OpenCV nous servira également pour accéder à la webcam (ou aux webcams). De plus cette librairie capturera des images et les enverra à la librairie pour le traitement. Cette partie sera bien sûr recouverte par une partie écrite en C++ avec Qt pour lister les webcams disponibles par exemple.

Le développement se fera complètement sous Linux, avec à terme des essais sur d'autres systèmes d'exploitation pour porter l'application sous ces systèmes.

5.4 Étapes

Détail des étapes avec fonctionnalités pour chacune d'elle.

6 Conclusion

Conclusion (1 ou 2 paragraphe ?)