Cahier des charges

Algorithme MPDIR

Révision	Référence	Date
01	L-B-ESCPI-MPDIR-CDC	19/11/2012

Table des évolutions

Révision	Auteur	Date	Commentaire
01	mickael@leka	19/11/2012	Création du document

Table des matières

Introduction

Définition du document

Le présent document définit le cahier des charges concernant l'algorithme de traitement d'une image infra-rouge pour le compte de Leka.

L'algorithme recevra en entrée une image provenant d'une caméra infra-rouge et la traitera pour en ressortir des points d'intérêts. Ces points d'intérêts représentent la proximité de doigts enregistrés par une caméra infra-rouge.

Définitions techniques

Boost	Librairie C++ mettant à disposition une implémentation de divers algorithmes. *http://www.boost.org*
C++ SL	C++ Standard Library. Collection de fonctionnalités étendant le langage C++. http://www.cplusplus.com/reference
CppUnit	Framework de test. *http://sourceforge.net/apps/mediawiki/cppunit/index.php*
Debian	Système d'exploitation libre basé sur le noyau GNU/Linux. http://www.debian.org
gcc	Compilateur libre sur système GNU/Linux. http://gcc.gnu.org
HippoMocks	Framework de mocking, utilisé pour les tests unitaires et en collaboration avec un framework de test. https://www.assembla.com/spaces/hippomocks/wiki
Valgrind	Logiciel de profilage de programme. Permet le débugge, vérification d'initialisation des variables, la détection de fuites mémoire. http://valgrind.org

Documents référencés

Index	Référence	Description
[R1]	L-DI-RDC-CPP	Règles de codage C++ leka.

Contexte logiciel

Compilation

L'algorithme doit être développé dans un environnement linux basé sur **Debian**.

Le compilateur utilisé, pour compiler le code source de l'algorithme et ses composants, sera **gcc** dans sa version 4.6.

Précision

Pour la précision des données, on se contentera au maximum d'utiliser des types encodés sur 4 octets.

Le type « double » doit donc être proscrit.

Librairies

Algorithme

L'algorithme de traitement ne doit dépendre que des librairies suivantes :

- **C++ SL** dans sa version implémentée par **gcc**4.6
- **Boost** dans sa version 1.52.0

Tests unitaires

Les tests unitaires devront utiliser les librairies suivantes :

- **CppUnit** dans sa version 1.10.2
- **HippoMocks** dans sa version directement récupérable à l'adresse suivante : *https://www.assembla.com/code/hippomocks/git/nodes/master/HippoMocks/hippomocks.h*

Tests d'intégration

Les tests d'intégration utiliseront la librairie **CppUnit** dans sa version 1.10.2. D'autres librairies peuvent être utilisées pour faciliter la mise en place des tests.

Pour chacune des librairies utilisées, il est demandé de fournir :

- La librairie elle-même ou un lien de téléchargement
- Sa description
- L'utilisation qui en est faite pour chacun des tests d'intégrations

Contexte matériel

L'algorithme doit pouvoir tourner au mieux sur un processeur ARM mono-cœur cadencé à 700MHz en utilisant au plus 50% des ressource disponibles.

Développement

Projets

Le développement de l'algorithme sera décomposé sous forme de trois projets :

- Un projet générant une librairie statique de l'algorithme
- Un projet de tests unitaires validant la librairie statique de l'algorithme. Ce projet produit un exécutable lançant tous les tests unitaires de l'algorithme.
- Un projet de tests d'intégration produisant un exécutable lançant les tests d'intégrations de l'algorithme.

Bonnes pratiques

Les règles de codage leka [R1] doivent être respectées.

Les spécifications fonctionnelles devront être retracées dans une matrice de traçabilité.

Tests

Tests unitaires

Les tests unitaires doivent couvrir toutes les méthodes de toutes les classes composant l'algorithme.

La couverture du code doit être de 100%. Chaque partie non couverte doit être justifiée.

Tests d'intégration

Les tests d'intégration doit valider et faire référence aux spécifications fonctionnelles.

Scripts de lancement des tests

Plusieurs scripts de lancement sont attendus :

- Un script de lancement des tests unitaires.
- Un script de lancement des tests d'intégrations.
- Un script de lancement de la vérification des fuites mémoires.
- Un script de lancement de la vérification de la couverture de code.
- Un script qui lance tous les scripts précédents.

Utilisation de la mémoire

L'algorithme ne doit comporter aucune fuite mémoire.

Les fuites mémoires seront détectées par l'utilisation de **Valgrind** sur l'exécutable des tests unitaires.

Arborescence

Les projets respecteront l'arborescence suivante, à la casse près :

Dossier	Description
bin	Contient les dossiers des exécutables.

bin/algo-mpdir/debug	Contient les exécutables compilés en mode debug.
bin/algo-mpdir/release	Contient les exécutables compilés en mode release.
docs	Contient les dossiers contenant les documentations des projets.
docs/algo-mpdir	Contient les documentations de l'algorithme.
lib	Contient les dossiers des librairies.
lib/debug	Contient les librairies compilées en mode debug.
lib/release	Contient les librairies compilées en mode release.
projects	Contient les dossiers des Makefile des projets.
projects/algo-mpdir	Contient le Makefile de l'algorithme.
projects/algo-mpdir-tu	Contient le Makefile des tests unitaires de l'agorithme.
projects/algo-mpdir-ti	Contient le Makefile des tests d'intégration de l'algorithme.
src	Contient les dossiers contenant les sources des projets.
src/algo-mpdir	Contient les sources de l'algorithme.
src/algo-mpdir-tu	Contient les sources des tests unitaires.
src/algo-mpdir-ti	Contient les sources des tests d'intégration.
tests	Contient les dossiers contenant les scripts de test.
tests/algo-mpdir	Contient les scripts de lancement définies dans « Scripts de lancement des tests ».
tests/algo-mpdir/data	Contient les dossiers contenants les ressources nécessaires pour les tests d'intégration.
tests/algo-mpdir/data/test-n	Contient les ressources associées au test « n ». « n » étant le nom du test.
tests/algo-mpdir/reports	Contient les rapports suivants : - Rapport des tests unitaires - Rapport des tests d'intégration - Rapport de la couverture du code - Rapport des détections de fuites mémoire

Nom des fichiers

Le nommage des fichiers sources et leur emplacement doit suivre les instructions contenues dans les règles de codage [R1].

Les libraires statiques de debug et release doivent avoir, respectivement, les noms **algo-mpdir-dbg.a** et **algo-mpdir.a**.

Les exécutables de tests unitaires et d'intégration doivent avoir, respectivement, les noms **algo-mpdir-tu** et **algo-mpdir-ti**.

Les scripts de lancement des tests unitaires, des tests d'intégration, de la vérification de la couverture de code, de la vérification des fuites mémoire et du lancement de tous les scripts doivent avoir, respectivement, les noms **algo-mpdir-tu.sh**, **algo-mpdir-ti.sh**, **algo-mpdir-cc.sh**, **algo-mpdir-ml.sh** et **algo-mpdir-all.sh**.

Spécifications de l'algorithme

Description

L'algorithme s'initialise avec des paramètres de configuration.

Il reçoit ensuite en entrée une image provenant d'une caméra infra-rouge. Cette image est composée de tâches. C'est un tableau de points dont chaque point a une intensité.

Les doigts touchant la surface sont donc représentés par des tâches plus ou moins intenses. Ces tâches doivent donc être repérées sous forme de groupes et un groupe représentera un doigt touchant la surface tactile. La position du groupe sera calculée par son centre.

L'algorithme renverra ensuite la position des groupes sous forme d'association (numéro, coordonnées) où le numéro identifie un groupe et les coordonnées de sa position.

L'algorithme aura aussi une notion de mémoire. C'est-à-dire que pour une deuxième image traitée, si un groupe est détecté proche d'un groupe précédent, il aura le même identifiant que sur l'image précédente.

Les identifiants des groupes sont uniques. C'est-à-dire que si un groupe vient à disparaitre et à réapparaitre au même endroit, il aura un identifiant différent.

L'algorithme pourra être remis à zéro au niveau des identifiants.

Les paramètres de l'algorithme peuvent changer entre deux traitements d'images.

Interfaces

Les interfaces suivantes sont fournies dans des fichiers séparés :

Fichier	Description
Int2	Interface de définition d'un couple d'entier.
Image.h	Interface de définition d'une image.
ImageIR.h	Interface d'un objet de type image infra-rouge.
AlgoMPDIR.h	Interface de l'objet représentant l'algorithme de traitement de l'image infra-rouge.

Paramétrage

L'algorithme comporte plusieurs paramètres, à son initialisation, qui sont les suivants :

- threshold : seuil de prise en compte d'une valeur d'intensité, si la valeur est en dessous, elle est ignorée.
- maxElementSize : la taille maximum qu'un groupe peut avoir.
- maxElementNumber : le nombre maximum de groupes à détecter.

Entrées / Sorties

Spécifications fonctionnelles

Les spécifications fonctionnelles suivantes décrivent les attentes de l'algorithme.

REQ_ALGO_MPDIR_00010_A

L'algorithme s'initialise avec les paramètres de configuration threshold, maxElementSize et maxElementNumber.

REQ_ALGO_MPDIR_00020_A

Les pixels de l'image ingra-rouge ayant une intensité inférieure au seuil threshold doivent être ignorés.

REQ_ALGO_MPDIR_00030_A

L'algorithme doit renvoyer la position des groupes dans la limite de maxElementNumber.

REQ_ALGO_MPDIR_00040_A

A la détection des groupes, ceux qui ont l'intensité la plus grande et la taille se rapprochant le plus de maxElementSize sont pris en compte en priorité.

REQ_ALGO_MPDIR_00050_A

Si un ou plusieurs groupes supplémentaires sont détectés alors que le nombre de groupes au total, maxElementNumber, a été atteint, alors ces groupes supplémentaires sont ignorés.

REQ_ALGO_MPDIR_00060_A

L'algorithme doit renvoyer la position des groupes sous forme d'une liste d'associations (numéro, coordonnées).

REQ_ALGO_MPDIR_00070_A

Un numéro identifie de manière unique un groupe.

REQ_ALGO_MPDIR_00080_A

Si au traitement n d'une image un groupe est trouvé à une position $P_1(x_1, y_1)$ et qu'au traitement n+1, un groupe est trouvé à la position $P_2(x_2, y_2)$ et que $\|P_1 - P_2\| \leq \frac{maxElementSize}{d}$ alors le groupe est considéré comme étant le même et doit avoir le même identifiant.

d est une constante définit au sein de l'algorithme. On pourra prendre $d = 2$ temporairement.

REQ_ALGO_MPDIR_00090_A

Les identifiants peuvent être remis à zéro.

REQ_ALGO_MPDIR_00100_A

Les paramètres de l'algorithme peuvent changer entre deux traitements d'images.