INFORME

# TRABAJO PRÁCTICO FINAL

March 15, 2017

Baruffaldi Juan Manuel
Análisis de Lenguajes de Programación
Licenciatura en Ciencias de la Computación

# Contents

# INTRODUCTION

Se realizó una adecuación de librería HOpenCV de Haskell para utilizarse con las nuevas versiones de OpenCV 3.x. Inicialmente HOpenCV es compatible con la versión 2.0 de OpenCV. HOpenCV es un binding de OpenCV en C++ por medio de Foreing Function Interface en Haskell. A su vez los creadores de HOpenCV iniciaron un proyecto llamado CV-Combinator para generar funciones de alto nivel que utilicen las funciones de HOpenCV.
En este trabajo se combinaron las dos librerias en una sola, combinando la posibilidad de utilizar las funciones de bajo nivel con funciones de alto nivel que abstraen las estructuras de la librería. Se terminó el desarrollo de los módulos de HOpenCV para la nueva versión de OpenCV,se incorporó manejo de errores, un makefile de compilación limpio, se agregaron 3 módulos más para las funciones de alto nivel, 3 ejemplos de uso y la documentación de la librería generada por haddock.

## OpenCV

OpenCV es una biblioteca libre de visión artificial originalmente desarrollada por Intel. Desde que apareció su primera versión alfa en el mes de enero de 1999, se ha utilizado en infinidad de aplicaciones. Desde sistemas de seguridad con detección de movimiento, hasta aplicaciones de control de procesos donde se requiere reconocimiento de objetos.
Esto se debe a que su publicación se da bajo licencia BSD, que permite que sea usada libremente para propósitos comerciales y de investigación con las condiciones en ella expresadas.

Open CV es multiplataforma, existiendo versiones para GNU/Linux, Mac OS X y Windows. Contiene más de 500 funciones que abarcan una gran gama de áreas en el proceso de visión, como reconocimiento de objetos (reconocimiento facial), calibración de cámaras, visión estérea y visión robótica.

El proyecto pretende proporcionar un entorno de desarrollo fácil de utilizar y altamente eficiente. Esto se ha logrado realizando su programación en código C y C++ optimizados, aprovechando además las capacidades que proveen los procesadores multinúcleo. OpenCV puede además utilizar el sistema de primitivas de rendimiento integradas de Intel, un conjunto de rutinas de bajo nivel específicas para procesadores Intel.

## FFI

La Interfaz de Función Extranjera (Foreing Function Interface) permite a los programas de Haskell cooperar con programas escritos con otros idiomas. Los programas Haskell pueden llamar a funciones extranjeras y las funciones externas pueden llamar al código Haskell.
La interfaz de función externa (FFI) es una extensión del estándar Haskell.
Pero cuando se empieza a trabajar con datos retornados por las funciones C, estas son devueltos en forma de estructuras o punteros de C. Para manejar correctamente esto es útil usar el programa hsc2hs.

## Comando hsc2hs

El comando hsc2hs se puede utilizar para automatizar algunas partes del proceso de escritura de enlaces Haskell al código C. Lee una fuente casi Haskell con construcciones especiales incrustadas y genera un archivo Haskell real con estas construcciones procesadas, basándose en información tomada de algunos encabezados C. Las construcciones adicionales tratan de acceder a los datos C de Haskell.

También puede emitir un archivo C que contiene funciones C adicionales que se van a vincular al programa, junto con un encabezado C que se incluye en el código C al que se compilará el módulo Haskell (cuando se compila a través de C) y al archivo C.

En realidad, hsc2hs no genera directamente el archivo Haskell. Crea un programa C que incluye los encabezados, se compila automáticamente y se ejecuta. Ese programa produce el código Haskell.

"Haskell file" es el archivo Haskell después de que ghc lo ha compilado en C (es decir, un archivo .hc), "C program" es el archivo de Haskell. Programa que genera el archivo Haskell, "archivo C" es el archivo C generado opcionalmente, y "encabezado C" es su archivo de encabezado.

## HOpenCV

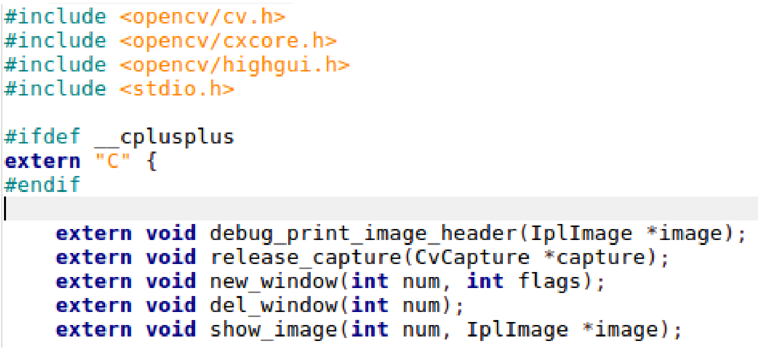
OpenCV bindings para Haskell de bajo nivel. Las funciones de alto nivel basadas en esta libreria se encuentran en el paquete cv-combinators.
La documentación del módulo en línea se encuentra caida, pero se podía encontrar en http://www.ee.bgu.ac.il/ noamle/.
El binding se realiza a través de un Objeto en C++ por medio de la librería Foreing Function Interface de Haskell.

## CV-Combinators

Es un wrapper de alto nivel, funcional, basado en el paquete HOpenCV como backend. CV-Combinators proporciona una biblioteca combinatoria funcional para visión por computador, basada en el paquete de procesador asignado.

# DESCRIPCIÓN DEL PROYECTO

Se continuó el desarrollo de la librería HOpenCV, extendiendo sus modulos base y actualizando la compatibildiad de OpenCV 2.0 a OpenCV 3.1. Se desarrolló un modulo de manejo de errores y ejemplos de uso.
La necesidad de esto surge al querer usar e instalar esta librería y no lograrlo de una forma sencilla (incompatibildiades viejas). Para llevar adelante el proyecto se realizó un binding a OpenCV 3.1 en c++ utilizando un objeto generado de la compilación de un archivo wrapper que importa las funciones de la libreria en c++. Se utilizó la libreria FFI de Haskell para manejar las llamadas al objeto y definir las funciones en Haskell que luego traducen la llamada.
Se trabajó sobre un archivo llamado "HOpenCV_wrap.cpp" diseñado para ser intermediario entre OpenCV y Haskell, pudiendo escribir algunas funciones en c++ que faciliten la llamada desde Haskell. Al compilar con las banderas de OpenCV con g++ se genera el objeto "HOpenCV_wrap.o"; con el cual intereactua las funciones de la librería FFI.
Para lograr compilar correctamente el archivo se incorporó dentro del header "HOpenCV_wrap.h" las siguientes isntrucciones:



```
#include <opencv/cv.h>
#include <opencv/cxcore.h>
#include <opencv/highgui.h>
#include <stdio.h>

#ifdef __cplusplus
extern "C" {
#endif

    extern void debug_print_image_header(IplImage *image);
    extern void release_capture(CvCapture *capture);
    extern void new_window(int num, int flags);
    extern void del_window(int num);
    extern void show_image(int num, IplImage *image);
```

Logrando incluir los headers de OpenCV y definiendo funciones intermedias en el archivo "HOpenCV_wrap.cpp".

Se generaron archivos .hsc para que por medio del programa hsc2hs se generen los archivos .hs que finalmente utiliza la librería. Dentro del Makefile se encuentran las compilaciones de g++, hsc2hs y ghc para la compilación de la librería completa con sus respectivas banderas de compilación.

La sintaxis de FFI es la siguiente para los archivos .hsc:

```haskell
{-# LANGUAGE ForeignFunctionInterface #-}

import Foreign
import Foreign.C.Types

foreign import ccall unsafe "stdlib.h rand"
     c_rand :: IO CUInt

foreign import ccall "stdlib.h srand"
     c_srand :: CUInt -> IO ()
```

En nuestro caso se utilizó llamadas a las funciones directamente de OpenCV:

```haskell
foreign import ccall unsafe "opencv/highgui.h cvConvertImage"
  c_cvConvertImage :: Ptr Priv_IplImage -> Ptr Priv_IplImage -> CInt -> IO ()
```

Y otras a las funciones del wrapper intermedio:

```haskell
foreign import ccall unsafe "HOpenCV_wrap.h &release_capture"
  cp_release_capture  :: FunPtr (Ptr Priv_CvCapture -> IO ())
```

Se incorporó en el desarrollo de funciones un módulo para menejo de errores por mónadas y luego se capturaron los errores de las funciones de OpenCV para mostrarlos por pantalla.

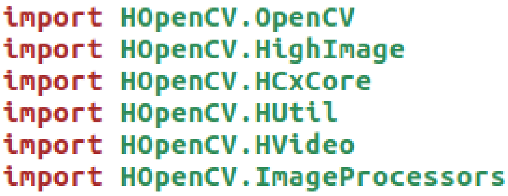
Esto se hizo para los módulos base (de bajo nivel). Luego para los módulos de alto nivel se desarrollaron funciones que llaman a las funciones base y generan excepciones para el manejo de errores manteniendo la sintaxis de HOpenCV. Se sumó el desarrollo de "ImageProcessors" creado para CV-Combinators como funciones de alto nivel y se testeó por medio de "Test.hs" obtenido de CV-Combinators.

Luego para testear y entender el uso de la librería se crearon 3 ejemplos de uso que se pueden encontrar dentro de la carpeta "examples" en el directorio de la librería.

## MANUAL DE USO E INSTALACIÓN DE SOFTWARE

Se desarrollo un Makefile que compila todos los archivos con sus respectivas banderas de compilación para facilitar el instalado de la libreria. A su vez se desarrollaron algunos ejemplos que muestran la forma de utilización de la libreria.

A la hora de utilizar la librería se deben importar los siguientes módulos, dependiendo el conjunto de funciones de OpenCV que se desea utilizar:



```
import HOpenCV.OpenCV
import HOpenCV.HighImage
import HOpenCV.HCxCore
import HOpenCV.HUtil
import HOpenCV.HVideo
import HOpenCV.ImageProcessors
```

Y para compilar utilizamos el siguiente comando:

```
ghc –make file.hs HOpenCV/CV/HOpenCV_wrap.cpp ‘pkg-config opencv –cflags –libs‘ -o file
```

Dentro de la carpeta "examples" se encuentra un Makefile para compilar los ejemplos.

## ORGANIZACIÓN DE ARCHIVOS

Los archivos mantienen la siguiente estructura de organización:

HOpenCV-0.5.0.1/

- Archivos "README y otros archivos"
- dist/
- doc/ "Documentación de la librería"
  - haddock/ "Documentación en html"
- src/ "Código fuente de la librería"
  - examples/ "Archivos de ejemplo de uso de la librería"
  - HOpenCV/ "Módulos de alto nivel y de bajo nivel"
  - Archivos "Módulos de alto nivel"
  - CV/ "Módulos de bajo nivel"

## DECISIONES DE DISEÑO

Se continuó el desarrollo de HOpenCV pero adaptando la librería a OpenCV 3.x. Se decidió solamente trabajar con los modulos base.

Se incorporó manejo de errores a través de una monada ErrorCV dentro de los modulos basicos de bajo nivel.

Se decidió combinar la librería cv-combinator dentro de HOpenCV para facilitar el uso creando funciones de alto nivel. En este caso las funciones generan excepciones para capturar los erorres de los modulos básicos. Se mantuvo la sintaxis de las funciones y se sumo el trabajo de cv-combinators dentro de la libreria.

Se utilizó haddock para la generación de la documentación de funciones.

# BIBLIOGRAFÍA

Anand, U., 2010. The Elusive Free Radicals, *The Clinical Chemist,* [e-journal] Available at:<`http://github.com/sinelaw/HOpenCV.git`> [Accessed 2 November 2013]