COLUMBIA UNIVERSITY
The Fu-Foundation School of Engineering and Applied Science

COMS W4115: Programming Languages and Translators

By Professor Stephen Edwards

simplified Video and Image Processing(sVIP)

Project Proposal

Vaibhav Jagannathan (vj2192@columbia.edu)

Shubanshu Yadav (sy2511@columbia.edu)

Bhargav Sethuraman (bs2814@columbia.edu)

September 25, 2013

Objective

This programming language aims to provide an environment to read, write and revise with ease as compared to other conventional programming languages. There are two primary functions that our language can perform.

1. Create videos from frames/image.
2. Alter images by resizing/filtering/modifying formats and perspectives.

Introduction

In recent times Image processing has become a very popular topic. A number of amateurs and dilettantes wish to try their hand at image processing. Some discouraging factors in the currently available image processing applications are the extremely expensive licenses and counter-intuitiveness of the programming languages.

More and more people are using photos as an essential part of their diaries or blogs, thanks to the widely prevalent digital cameras that gained popularity since year 2000.We aim to create a simple, easy to use, language that will provide people with the means to process images and videos. This involves modifying images/videos, appending videos with frames, storing, etc.

Syntax

Data types:

DATA TYPES	EXAMPLE DECLARATIONS
Integer	integer a = 0;
Float	float fl = 144.65;
Image	Image I = image("C:\..\..\Path"); Image I1 = I2;
Video	Video v = video("C:\..\..\Path"); Video v1 = v2;
Bool	bool b = true;
Arrays	Any of the above data types can be used as arrays in one form or the other as shown below : integer a[10]; - array with 10 integer elements

Operator:-

Operator	Description
+ -	Used for the Addition and subtraction of the images, videos, arrays and integers.
*** /**	Used for multiplication and division of the data types
=	Assignment Operator
== !=	Compare whether the two images and videos are equivalent or not. Returns Boolean.

<>	Modify operator to convert an image/video to different formats such as YUV and RGB.
	Also used to resize images/videos.
	Example :
	Image Im;
	Im<>(YUV); //converts image to YUV format
	Im<>(640,480); //resizes Im to a 640 x 480 image

Built in functions

Operator	Description
Rotate	Takes 4 parameters – image/video, axis, offset from the origin for rotation and angle(degrees). Rotates the image by the specified angle around one of the specified axis. For a video, it rotates all frames. Example : Image I1; Rotate(I1,x,2,10); // Rotates I1 by 10 degrees around the axis x = 2.
Info	It accepts a single parameter – Image/Video It displays the size(in bytes) and dimensions(L x W in terms of pixels) for an image. For a video it additionally displays the number of frames.
Show	Displays the image, the integer, et al.

Simple sample code snippet:

```
main :
Image Im = Image("D:\Lena.png")
Video Vi;
Integer I;
for(I = 0; I < 360; I = I + 1)
{
Vi = Vi + Im;
Rotate(Im, x, 0, I); //Rotates the image along x axis.
}
Show(Info(Vi));    //Displays Vi size, dimension and number of frames
Vi<>(320,160);    //Resizes each frame in the video to the size 320 x 160
Show(Vi);          //Plays the video which now show one complete rotation of the image around the x axis
```