Abstract — This paper presents the 'on-line Image Processing' (OLIP) Toolbox (running on an Intel-x86 platform). The primary goal of this toolbox is to perform an on-line processing from images acquired from a frame grabber (the VG5-Scion is used). The Fourier and Wavelet transform are implemented; also Mathematical Morphology, Convolution, Correlation and Statistics tools are included. This toolbox was initially developed for educational purposes at the Pontificia Universidad Catolica (Peru). It also provides a simple way to add new tools and shows how to develop a PCI device driver to add different frame grabbers support.

Index Terms — Digital Library, Learning Models and Technology.

I. INTRODUCTION

A simulation process is use to find a satisfactory solution; the main drawback, in the case of digital image processing (DIP) is the time that this simulations can take (using a PC). They are very good simulations packages for DIP (i.e. Khoros [1], Matlab, Octave, etc.) with a very complete set of toolboxes, but normally the simulation process is somehow slow and, most important, these packages do not have an interface to use hardware to acquire directly signals or images (i.e. in the case of digital signal processing an interface with the sound device would be recommended, a project like describe above has been already developed [2]; in the case of DIP an interface with a frame-grabber or with a parallel-port digital camera is also needed), so the overhead (time) to perform a complete simulation is increased.

In our case, at the digital signal and image processing group (DSIPG) at the Pontificia Universidad Catolica from Perú (PUCP), the standard tool for image processing simulation that we use is Khoros; we also have a frame grabber (VG5-Scion [3]) but it came with a Windows-based driver. So we have an extra overhead to complete simulations; the solution was to develop a Linux-based driver for the frame grabber and also integrate this driver with a library (in both ANSI C and C++) for image processing, with an X windows viewer, that uses optimized algorithms to allow on-line processing and also the results are shown on-line; finally, this library was integrated to Khoros (here we must point out that this toolbox is autonomous from Khoros) so our simulation time is reduced.

We must remark that the license of this toolbox and the VG5-Scion driver for Linux are GNU-type and it could be download directly from the anonymous ftp server of the DSIPG (see III).

In section II the PCI driver for the VG5-Scion frame-grabber is described; its functionality is also described. In section III the tools for DIP are discussed; performance is also described here. In section IV the interface between this library and Khoros is discussed. In section V is shown how to use these tools as an ‘application-ready’ library. Finally, conclusions and future features of this library are described in section VI.

II. VG5-SCION FRAME GRABBER

A. Features

The VG5-Scion frame-grabber has nice features for acquiring digital image: it uses any PCI slot, has 8 bits per pixel (gray scale image), a maximum 768x512 resolution (CCIR), 1MB on board memory (two buffers). This frame grabber allow the user to load its own colormap directly to the on-board memory; also voltage levels for digitalization can be set via software. In our case we use this frame grabber together with a COHU CCD camera.

B. Driver

The PCI driver for this frame grabber was initially developed for 2.0.x Linux kernel; recently it was ported for 2.2.x Linux kernels. In [4] is describe how this driver was developed; a more general references on how to develop a device drive for Linux can be found in [5].

The main features of this driver (called vg5) are:

• The user can set image size.
• The user can set the colormap of the acquired image (normal, inverse or user-define colormap). This colormap is load on the HW of the frame grabber, so there is no overhead on the application program and is also useful for scenarios which have too much illumination (a logarithm colormap could be used) or...
with very bad illumination (an exponential colormap could be used).

- The user can set minimum and maximum voltage level of digitalization (this could be used for a HW equalization of the acquire image).
- Still on development, the user could set sampling frequency (of each frame) and the driver will write data to a circular buffer (in a similar way the sound driver from Open Sound System [6] works); the user would have to read from this circular buffer before data is overwritten.

III. MATHEMATICAL LIBRARY

A. Mathematical tools

A good library to perform mathematical operations must have two main characteristics: fast algorithm for different operations and a suitable interface to prototype new operations (new tools) and new operators (i.e. new kernels for convolution, correlation, mathematical morphology, wavelets, etc).

The OLIP library includes the following mathematical tools: FFT, discrete Wavelet transform (DWT), Convolution (linear), Correlation (linear), Mathematical Morphology (MM); erosion, dilation, opening and closing are supported and statistical tools (histogram, standard deviation, mean value, etc). These tools could be used in both an off-line and on-line fashion. In the first case an input image (file) is needed, the result is saved in an output file; In the second case, image are directly acquired from the frame grabber and the result is shown in the PC display; results could be also save to an output file.

In the case of OLIP library a special care was taken when choosing algorithms for the FFT [7] and DWT[8,9]. For the FFT the user can perform the analysis just for rows, columns or for both; results could be centered (DC at N/2, assuming input images is NxN), and could be scaled by unity, by 1/N or 1/sqrt(N). For the Wavelet transform user can define how many levels will be analyzed and in which fashion (standard or non-standard decomposition). The inverse transform is defined for both (Fourier and Wavelet, they are used in the off-line fashion; they could be used in the on-line fashion but it does not make any sense, even though inverse DWT could be use to extrapolate a short image acquired from the frame buffer.

If a user needs to use a particular kernel not supported, it could be pass like a parameter to the precompiled programs or could be add directly to the library (recompilation is needed in this case). For DWT the library supports several popular kernels (Haar, Daubechies, Splines, Symmlets, etc); in the case of MM it has two predefined structural elements (SE) predefined (cross and X 3x3 structural elements). Structural elements could be of any size, but they must be either binary or grayscale; for convolutions there are a few predefined kernels (i.e. Sobel X, sobel Y, etc.). In the case of Convolution and Correlation, kernels could be also of any size, and coefficients are always float type.

For visualization of images acquired from the frame grabber MIT shared memory extensions [10] for the X windows system is used (when available, if not standard X11 library is used); this is needed especially for large image (this is faster than using the standard X11 libraries).

This entire library could be accessed via anonymous ftp at the DSIPG FTP server: pds01.pucp.edu.pe in the pub/OLIP directory.

B. Performance

The time-performance of the compile programs, based on the OLIP library is presented here. All programs were tested on Intel Pentium based PC (200 MHz and 32MB RAM) running on Linux 2.2.2 box. For all cases describe below, gray scale image with a 512x 512 size were acquired using the VG5-Scion frame grabber; also, in all cases the resultant image is show in the display simultaneously.

- Video Visualization (no mathematical operation) 8.3 Frames / Sec.
- MM (erosion/dilation 3x3 SE) 4.2 Frames / Sec.
- MM (erosion/dilation 3x3 SE border identification) 3.5 Frames / Sec.
- MM (opening/closing 3x3 SE) 2.5 Frames / Sec.
- FFT 0.4 Frames / Sec.
- Convolution (3x3 kernel) 1.6 Frames / Sec.
- DWT 0.1 – 2 Frames / Sec. (Depends on the used kernel)
- Histogram 6.5 Frames / Sec.

IV. INTERFACE WITH Khoros

All the programs developed using the OLIP library can be used from any shell (using the command line) but all this programs need a set of parameters to be specified (width and height of the image, which colormap will be used; also, depending on the mathematical operation to be perform different options must be pass); it is also desirable that the OLIP library could have a larger database of algorithms for different simulations. From these stamens is conclude:

- A graphical user interface (GUI) is needed.
- A drawback of the OLIP library is that it does not provide all the tools necessary for digital image simulation.

The solution for both problems is to add the utilities of the OLIP library to Khoros, to ease the creation of GUI and to
have all the large tool support for image simulation: the result is the OLIP toolbox.

At this time results obtains from the OLIP toolbox are save in raw files (i.e. acquired image frames, the FFT, etc.) so Khoros could use this results without any problem, but not directly; an application from Khoros is use to convert data from the raw format to the KDF format (Khoros 2.0 data format). Because of this, results could be used from any other simulation software (Matlab, Octave, etc).

V. Application

At this moment an application based on this library is being develop; this application needs to measure the liquid level from a cola bottle (glass) in a cola factory. This is of vital importance because when the actual level of the liquid is out bounds there is a high probability that inside the bottle exists glass biroute due to explosion of any contiguous bottle in the line production.

At this time, this quality control is done by human visual inspection (an employee must check all bottles). This a very tedious work for a person (even though they are substituted every two hours) because they suffer from visual fatigue, because behind the bottles (in the line production) there is a high illuminated panel that make the worker to decrease his ability to distinguish color, so he just see in a black and white fashion (see fig. 1); when the work ‘thinks’ there is a ‘bad’ bottle, he must stop the line production, and carefully observe the bottle (about 10 to 20 seconds) after that he decides if it is a good bottle or a bad one. The average is that every 5 min. the line production must be stop.

In fig.3 is show the acquired image and in fig. 4 is show the border identification; the software is in a development stage, but we already have some significance results: the acquired image is 200x300 pixel (the minimum size needed); the system can process the input image at 4.5 frame per second (this is the average); the user could easily set the limits bounds (using the mouse) so every bottle that does not fit into this bounds will be reject.

VI. Future Features

The intention of the authors is that this library is developed as a dynamic library, so programs will not need to recompile with new versions of this library. Also source will be available.

This library could be add like a toolbox to other software simulations packages (Octave is the next possible candidate)

The use of parallel port digital cameras would add a high portability to this toolbox. Here we must point out that only Conectix Quick Cam and the so-called ‘egg-cam’ from Panasonic has support under Linux. It would be very easy for anyone who has this type of parallel port digital cameras to use them together with the OLIP library (they must only change the calls to the VG5 driver for the right calls that the new driver needs)

VII. Conclusions

On-line image processing using a PC is possible. Together with a low cost HW (like a parallel port digital camera) any PC could, running Linux, could be used as an interface to solve a real problem where DIP is needed.
Initially this library was written with educational purposes, but this could be used like an application ready library to solve real problems.

Because Linux is a truthfully preemptive operating system performance under this operating system is better than under Windows.

The time to complete a simulation processes is reduced because all the tools needed (acquisition, math tools with fast algorithm and a large number libraries) have been integrated; in this particular case, they have been added to Khoros (acquisition, and faster algorithms, when proper).

A frame grabber with DMA support is recommended (VG5-Scion frame grabber does not support DMA). Also writing a PCI driver to add different frame grabber will not be a difficult work when documentation is available.

**REFERENCES**


