Qvision

Qt's Image, Video and Computer Vision Library.

Artificial Perception Group
http://perception.inf.um.es

PARP research group
http://perception.inf.um.es

University of Murcia, Spain

Motivation

- Computer Vision research often needs prototyping.
- This implies dealing with many technical issues, from video input managing to graphical output image, including parallel programming.
- In Real time prototyping, performance needs are important
- Few tools/libraries offer an integral solution to all of these topics.

Solution

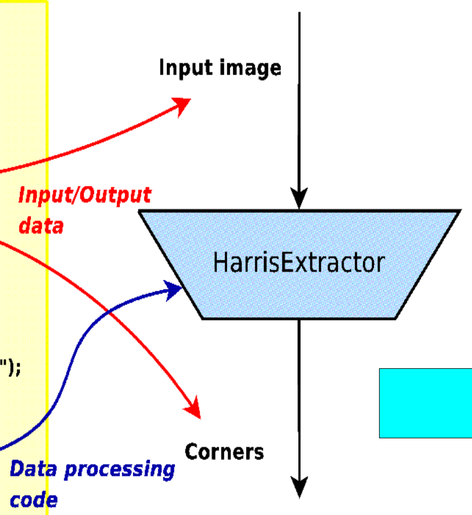
A development framework, QVision. It provides:

- Fast Computer Vision prototype development.
- Efficient performance on image processing and math functionality.
- Special focus on parallel programming and reusability.
- An Open Source solution, free to use, free to contribute.

1. Fast Computer Vision algorithm prototyping with QVision

1.1 Data processing components development

```
class HarrisExtractorWorker: public QVWorker
   {                                          (1)
   public:
      HarrisExtractorWorker(QString name): QVWorker(name)
         {
(2)      addProperty< QVImage<uChar,1> >("Input image", inputFlag);
         addProperty< QList<QPointF> >("Corners", outputFlag);
         }

      void iterate()
         {
         const QVImage<uChar> image =
                        getPropertyValue< QVImage<uChar> >("Input image");

(3)      QList<QPointF> hotPoints;
         // [ ... ]
         // Get Harris corners and store them in 'hotPoints' list
         // [ ... ]

         setPropertyValue< QList<QPointF> >("Corners", hotPoints);
         }
   };
```

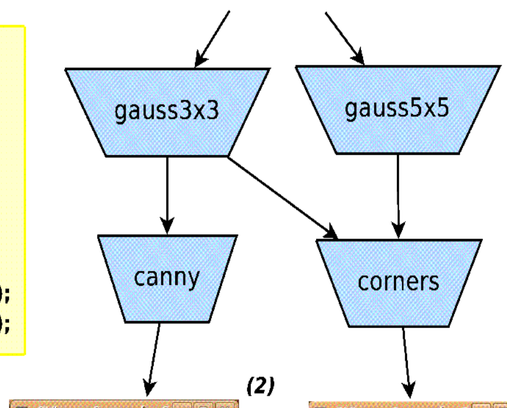


(1) Inherit from QVWorker class.
(2) Define input/output data in the class constructor.
(3) Define data processing code.

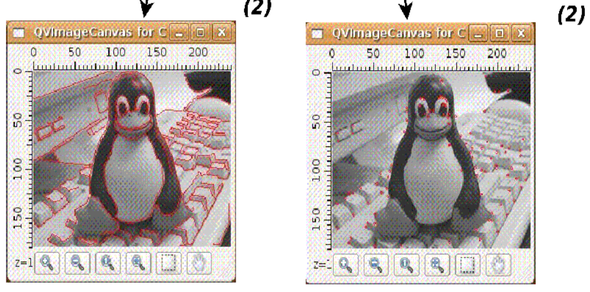
1.2 Components linking

```
QVMPlayerCamera   videoInput();
```


```
QVGausianFilterWorker gaus3x3(3), gaus5x5(5);
QVCannyWorker                 canny;
QVGaussDiffCornerWorker       corners;

videoInput.link(gaus3x3);
videoInput.link(gaus5x5);

gauss3x3.link("OutputImg", canny, "InputImg");
gauss3x3.link("OutputImg", corners, "InputImg");
gauss5x5.link("OutputImg", corners, "InputImg");
```

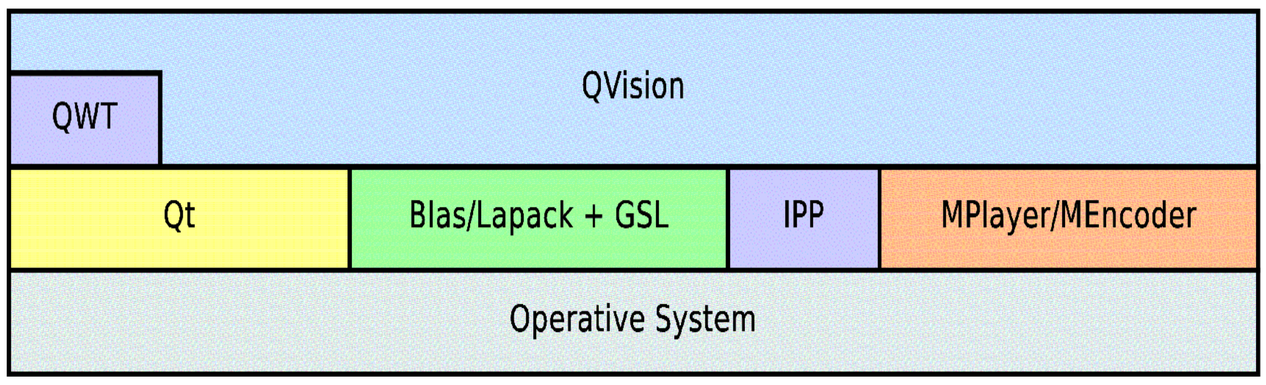



```
QVImageCanvas    cornersCanvas,
                 cannyCanvas;

cornersCanvas.link(corners);
cannyCanvas.link(canny);

...
QVApplication.exec();
```



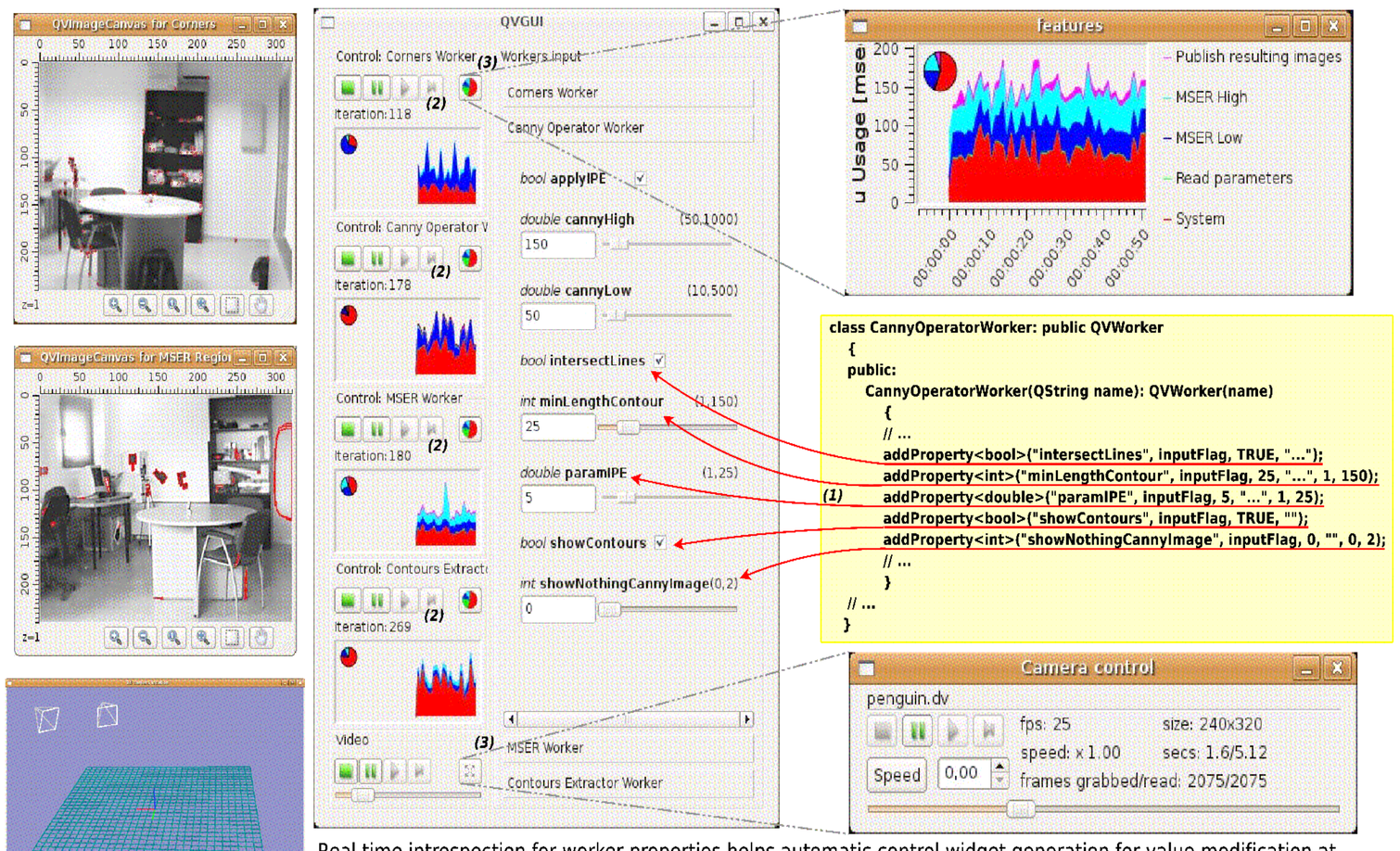
(1) Multiple video and data input types: webcams & digital cameras, video files, remote streams, etc...
(2) Out of the box image display widgets: zoomable, simple usage.

2. Comprehensive and efficient image processing + math functionality



- Wrapper functions for *Intel's Integrated Performance Primitives*.
- Math wrapper functions for *Blas/Lapack* and GNU's Scientific Library.
- Mplayer wrapper class *QVMPlayerCamera* provides capacity for reading from a wide spectrum of video inputs and formats.
- Computer Vision or relevant algorithms: *MSER*, *SIFT*, *SURF*, *RANSAC*, etc..
- Qt provides fast graphical user interface access, containers, and much more.

3. *Ad hoc* GUI interface and widget classes



Real time introspection for worker properties helps automatic control widget generation for value modification at execution time (1). Worker execution flow can be stopped, resumed or executed step by step thanks to these controls (2). The GUI offers triggers to open detailed CPU workload and video control menus (3). Also, QVision offers several data output widgets, such as image widgets (4), 3D scene canvases (5), etc...

4. Notes

- Worker software design pattern[2] is implemented in the library, which helps parallel programming and code reusability [1] with little expertise from the developer/researcher on these topics.
- Wrapper classes and functions for Intel's IPP, Blas/Lapack and GSL libraries look forward best time performance, embedded in object oriented programming syntactic sugar, and use ease.
- Code released under the GNU Lesser General Public License.
- Good tool for educational purposes.

5. References

- *[1] A Design Pattern for Component Oriented Development of Agent Based Multithreaded Applications. Euro-Par'08.*
- *[2] QVision, a development framework for real-time Computer Vision and Image Processing research. ICPV'08.*

This project has been supported by the Spanish MEC and European FEDER funds under grants *Consolider Ingenio CSD2006-00046* and *TIN2006-15516-C04-03.*