IPOL Editorial Committee Meeting

January 30th, 2014

Who Andrès Almansa, Miguel Colom, Mauricio Delbracio, Gabriele Facciolo, Rafael Grompone, Jose Lezama, Nicolas Limare, Enric Meinhardt, Pascal Monasse, Jean-Michel Morel, Laurent Oudre, Gregory Randall, Agustin Salgado, Eva Theumann

Where ENS Cachan, Cournot room 120

When January 30th, 2014, 12:00 to 14:15

Contents

1 IPOL workshops

There are multiple works, mainly web demos, on the IPOL development server `http://dev.ipol.im`. These works are often interesting or somehow useful but can not be published as-in in IPOL for multiple reasons: still in development, using unusual software components, not stable or documented yet. We want the possibility to make them publicly available and linked to the IPOL Journal.This is the concept of "IPOL workshops":

- a "WORKSHOPS" link is added to the IPOL menu bar, going to an index of workshops;
- every author/developer can freely decide to insert their works in the "workshops" list, by adding a link into the index.

Other possible benefits include: testbed for new kinds of data, interfaces, languages, etc.; use as tutorials, for education; showcase for conferences; collaborations with external (non-imaging) groups.

This has been implemented immediately after the meeting: the "WORKSHOPS" link goes to http://dev.ipol.im/ws/ and this page includes instructions for authors. Anyone with an account on the `purple` server can edit this index. Changes are saved every 5 minutes.

These workshops are closely related to the IPOL Journal (same people, same resources), but they are not part of the journal. There is no editorial control, no peer review, and no guarantee provided by the IPOL editorial committee on their quality or availability. As such, their appearance must not be misleading; workshops must not look like articles.

Three major issues remain; Nicolas Limare is in charge of solving them or proposing solutions:

- Access to workshops must be possible without password; the current solution (embedding the passwords in the link) is only a temporary fix.
- Workshop authors must have a solution to automatically start their own demos when the server is restarted, so that the service is always available.
- Workshop authors must have a solution to easily setup a redirection, so that visitors can access the service if they wish to move it to another location.

Maintenance of links to these workshops is a concern. With the current setup, a workshop is available as long as authors have an account on the development server and keep their demo running at the same address and their files published with the same file names. If a workshop becomes a published preprint or article, the server administrators can setup a redirection from the workshop URL to the preprint or article URL, on request.

A more powerful option is to maintain a list of permanent URLs (short addresses with a redirection, updated when the addressed content moves) which could be used for IPOL workshops as well as for any online content researchers need to mention in their works. This service is not, strictly speaking, part of the IPOL workshops, but both can integrate well. Rafael volunteered to manage this permanent URL system.

Next steps: inform purple users about usage and risks, let the workshops index fill up then publish a "news" item.

2 3D in IPOL

One article on 3D point clouds has been accepted, other articles are expected. IPOL want to be able to publish articles on 3D data. There are no obstacles for the article itself (text and code), but some work is needed for web demos. We need solutions for interactive rendering in a browser and we need to control the impact of large data (upload to the server, processing, and download to the browser).

Interactive rendering is possible with two JavaScript libraries: `three.js` for meshes and `XB-PointStream` for point clouds. These two solutions use the WebGL technology, and require recent and capable browsers and some hardware (GPU) support. This probably excludes Internet Explorer (12.7% of IPOL visits in 2013) and computers built before 2010.

There also are some challenges with the size of the data. Large data can not be easily uploaded to the demo; instead, input data proposed by the demo will be used. Large data may imply long processing; this can be mitigated by downsampling point clouds, or clipping meshes (when appropriate). Downloading the result files for rendering is also a potential bottleneck: a 50MB file will take some time to be available in the browser; on-the-fly compression (by the app server or front-end web server) can mitigate the issue, but is sensible to bad proxying (such as currently done on the ENS Cachan network).

Miguel Colom is in charge of this development. It will be tested in an IPOL Workshop before standard installation on the public server.

3 Audio in IPOL

Five articles are currently waiting in the IPOL review pipeline. They could be included in IPOL, at least as public preprints and to show it is possible. This may or may not be a replacement for the J-RASP (ex-SPOL) project, in case it does not come to life.

Some demos already exist with audio processing. They work as prototypes, can can already be inserted into the IPOL Workshops index. Their integration into the main demo server requires a little more development work:

- Input files must be accepted, converted and shortened if needed, from a few usual audio file formats (MP3, FLAC, WMA, OGG) to the file format read by the algorithmic code. This input processing must be done in the Python demo layer; it could use the `pysox` package, or other technical solutions.
- Output files should be usable in common browsers. The current solution (HTML5 audio tag) only works with MP3 files in Internet Explorer, and the MP3 compression could be a problem for some algorithms. One solution could be to simply add a link to the file for download instead of embedding it into the web page.

Some audio algorithms are implemented in MATLAB. Web demos could be built for these algorithms by using the methods tested and documented by Juan Cardelino, and tested as IPOL Workshops.

Then, if these works are to be published in IPOL and if IPOL wants to accept new audio processing submissions, new items must be added to the IPOL Software Guidelines and IPOL Author Manual:

IPOL Software Guidelines, 2.5 External Libraries ·
add the `sndfile` library, version 1.0.x, with a link to `http://www.mega-nerd.com/libsndfile/`

IPOL Software Guidelines, 2.7 Usage and Input/Output ·
add the "WAV for audio files" format, with a link to `http://www.digitalpreservation.gov/formats/fdd/fdd000001.shtml`

IPOL Author Manual ·
add an "Audio" section, after the "Pictures" section, with the instruction to limit audio files to 30 seconds and some details about copyright issues.

Someone must be in charge of this development. It will be tested in an IPOL Workshop before standard installation on the public server.

4 Video in IPOL

Like 3D and audio, one of the issues with video processing is the rendering in the browser. Showing a sequence of frames is possible, but rudimentary and not very adapted. One solution could be, like for audio data, to use a web viewer (HTML5 video tag or Flash player) on compressed video. A lightweight alternative is to let visitors download the data and view the files locally.

This would make video processing in IPOL easier, but the usual other issues remain: input processing and conversion, and computation time.

Someone must be in charge of this development. It will be tested in an IPOL Workshop before standard installation on the public server.

5 Demo System (re)Architecture

The current demo system is a monolithic server that takes care of everything: manage the HTTP communication, generate the HTML pages, receive and send the data and parameters to the

C/C++ algorithms, archive the results. This lack of modularity ties together the presentation step with processing, making them depending on each other. This makes difficult:

- changing the presentation interface with a different one;
- adding several presentation interfaces (2D, 3D, special cases, etc);
- creating external tools that could automate the creation of the demos without interfering with the demo system core;
- creation of APIs to interact with the server with a well-known format.

Separating the model (data/codes), view (web interface),and controller (workflow and processing) should be considered (in the future) for the scalability of the demo system. It can not be done as a part-time task by students or researchers focused on other works such as image processing research. It could be a good internship task for a software engineering student, and Miguel Colom is volunteering to coach such student. But we have no candidate so far.

6 Demo Generator

The "demo generator", written by Agustin Salgado, is a Python program with a web interface available in experimental form at `http://dev.ipol.im/~asalgado/ipol_demo_generator/`.

This program takes the description of an IPOL (source code location and compilation details, input and output, result formatting, etc.) from a web form, and saves all that into a single text file, following an ad-hoc language (demo description language, DDL). Then this text description of a demo can be used in two modes:

interpreter: In this mode, a web demo is run, following the specification. This is a dynamic interpretation of the DDL file, and can be used to test the specification.

compiler: In this mode, the Python code is generated for a demo following the DDL file specs. This code can then be installed as a stand-alone demo service, for further developments or for public release.

For the moment, some restrictions apply: input data must be already available and identified on the server; demos created with this system are restricted to the standard workflow (input-params-wait-result). Seven types of demos have been identified, four of them already proposed as standard templates.

Some development and testing work is still needed before this system can be used "for real".

7 Control and Validation of the Demos

There is an ambiguity about who is in charge of checking that demos are correctly written: reviewers are not supposed to check it; editors in charge of a submission are often not able or not willing to do it. Doing it just when we receive them for integration and publishing is time-consuming and can delay a lot the public availability of a demo, resulting in preprints published without demos or articles waiting for weeks after being accepted before they become publicly available.

This is even more difficult when demo authors wrongly believe that "*if it seems to works on the development server, then everything is OK and it can be installed as-is on the public IPOL demo server.*"

Three solutions are proposed:

- Some better "demo guidelines" are needed. Nicolas and Miguel will prepare a list of precise instructions for the demo authors. Then this list will be communicated to the person in charge of creating the demo by the editor.

- A restricted test environment would help. Available from the development server, it would exactly mimic the software environment of the production server. Nicolas will work on it, but it will probably not be available before a few months.
- Some members of the board are able to check the quality of a demo and are willing to help: Mauricio, Miguel, Rafael, Enric, Jose. They can receive a few requests for verification from Miguel and perform a preliminary screening.

We also note that the demo generator will help because it will generate only standardized code.

8 Other Topics

These topics were not discussed, for lack of time. There are mentioned hereafter for future reference.

8.1 IPOL Common Core

from Jérôme Gilles, edited – It will be extremely valuable to have a basic set of functions/data structures that all developers must use (like images structures with their macros, reading/writing functions, basic arithmetic functions, ...). For example, I was using my own code to solve a model and I also used functions written by [author A]. One reviewer asked why I was not using the functions developed by [author B] to solve the model. Code from [author B] was using some specific data structures, [author A] was also using his own, so finally the code would contain more lines only dedicated to convert data between the two formats, so I refused the change. I believe it would be better if everybody was using the same data structure definition and a standard set of basic function. IPOL would gain in interoperability between the different proposed algorithms and would facilitate the developers' work.

8.2 About Selling IPOL

from Gabriele Facciolo, José Luis Lisani and Bertrand Kerautret, edited – Showing the benefits of IPOL for the community is not enough. To "sell" IPOL we must advertise the benefits of using it for the individual and the laboratory where they work.

Success Stories We need comparative figures about paper citations, faster adoption/acceptation, grants and funds.... This info should be publicly available and included in future IPOL presentations.

Lab Notebook IPOL is also a tool to track research and collaborate. We could "sell" (even commercially) IPOL as a private workgroup platform.

Benchmarks We're already positioning IPOL as platform to compare state of the art algorithms (a replacement for the Middleburry benchmark, for example). In the future, when the critical mass is reached, we could just say it is a way to benchmark algorithms. But this is not the case yet.

Workshops To lower the pressure of writing a paper and get faster to the critical mass, we could re-introduce the workshops. Allowing authors to pre-publish algorithms without a description. Demos without the text description also helps authors submit their work to a journal or conference. The geometry group is also interested and could do it on their new IPOL server, recently purchased and installed.

8.3 Future Articles

from Jean-Michel Morel, edited – To be recognized, IPOL needs to collect the state of the art in multiple image processing topics. Editors are invited to take a topical section in their hands and propose a publishing plan to collect this state of the art. Workshops, as discussed, can help us reuse other codes already available elsewhere.