Inviwo — A Visualization System with Usage Abstraction Levels

Lucas Zamprogno

More info, video, and more at https://inviwo.org/
Motivations

- Accessibility: Should be accessible to those without programming experience
- Performance: Take advantage of low-level optimizations
- Adaptability: Work with many algorithms and technologies at once
Design Principles: Interactive Development

- Want to be able to view low-level change impacts
- Requires recompiling and/or observing source code changes
- Observe changes throughout pipeline
Inviwo Implementation: Interactive Development

- Integrated editor allows for low-level code changes that are immediately reflected in the output
Design Principles: Debugging

- Typically challenging to debug a multi-stage pipeline
- Idea: Have “ports” in and out of each stage to view intermediate data
- Different data types can support different views
Inviwo Implementation: Debugging

- Inspect output from each processor in the pipeline
Design Principles: Documentation

- Tends to be targeted at developers, but vis designers need access
- Suggest incorporating it into the API designers use
- Tailor documentation appropriately
Inviwo Implementation: Documentation

```cpp
/**
 * \docpage{org.inviwo.Background, Background}
 * ![()]\docimage{org.inviwo.Background.png?classIdentifier=org.inviwo.Background}
 * Adds a background to an image.
 * The following mixing is applied
 * \[
 * \text{out.rgb} = \text{in.rgb} + \text{color.rgb} \times \text{color.a} \times (1.0 - \text{in.a})
 * \text{out.a} = \text{in.a} + \text{color.a} \times (1.0 - \text{in.a})
 * \]
 * ### Imports
 * * __ImageInport__ Input image.
 * *
 * ### Outputs
 * * __ImageOutport__ Output image.
 * *
 * ### Properties
 * * __Style__ The are three different styles to choose from Linear gradient, uniform color, or checker board.
 * * __Color1__ Used as the uniform color as and color 1 in the gradient and checkerboard.
 * * __Color2__ Used as color 2 the gradient and checkerboard.
 * * __Checker Board Size__ The size of the rectangles in the checker board.
 * * __Switch colors__ Button to switch color 1 and 2.
 *
 */

*brief* Adds a background to an image.

```

```cpp
class IVW_MODULE_BASE_API Background : public Processor {
public:
    Background();
    virtual ~Background();
```

```

---

**Background**

- **Imports**
  - * ImageInport Input image.

- **Outputs**
  - * ImageOutport Output image.

- **Properties**
  - * Style* The are three different styles to choose from Linear gradient, uniform color, or checker board.
  - * Color1* Used as the uniform color as and color 1 in the gradient and checkerboard.
  - * Color2* Used as color 2 the gradient and checkerboard.
  - * Checker Board Size* The size of the rectangles in the checker board.
  - * Switch colors* Button to switch color 1 and 2.
Design Principles: Testing

- Unit testing often used for low-level code
- Open that option up to high-level designers
- Need to compensate for hardware differences
Inviwo Implementation: Testing
Design Principles: Interoperability

- For performance reasons, want access to computing platforms like OpenGL, OpenCL, CUDA
- Algorithms in one system can’t easily interact with others
- This can be a challenge with new technology not being compatible with existing algorithms
Inviwo Implementation: Interoperability

- Different computing systems can be chained together
- Get the low level optimizations of each
Inviwo Demo Video

https://www.youtube.com/watch?v=9yZWjxlV6OQ
Highlights

- Already shows good adoption
- Open source
- Extensible and user friendly
Highlights

● Supports software engineering good practices:
  ○ Debugger support
  ○ Documentation integration
  ○ Unit and integration/regression testing support
Paper Critiques

- Almost no discussion of their own limitations
- Seemed to oversell some points
  - Default implementation of port debugging
  - Running Inviwo tests on the same machine
Questions/Discussion