SimpleITK Survey:
Preliminary Results
General Statistics

• 253* participants

• Non-C++ users:
  – 21.34% (54 participants) have never used or don’t feel comfortable using C++
    • Specific distribution: 16 + 38
  – 51% (28) of non-C++ users have used ITK
  – 26 have never used ITK

• Non-ITK users:
  – 17.78% (45 participants) have never used ITK
  – 207 have used ITK
  – 44.44% (20) of non-ITK users are C++ users

*as of 11/6/10
Survey: Target Audience

1. ITK user list
2. VTK dev/user list
3. Orfeo toolbox
4. Slicer
5. R imaging group
6. NAMIC
7. UNC Medical Imaging
8. UNC Computer Vision
9. SCI – U. Utah
10. Image World
11. V3d-workgroup
12. SIIM workshop
13. Yale (R. Papademetris)
14. JHU (R. Taylor et al.)
15. Image Processing (CC/NIH)
16. C++ users on LinkedIn
17. Medical Informatics on LinkedIn
18. etc....
24.7% of the participants belong to academia
Population: Other

- Neuroimaging
- Ophthalmology
- Remote Sensing
- Statistics
- Microstructure modeling
- fMRI, DWI
- Historical paintings & preservation
- Radiotherapy
- Oceanography / Meteorology
- etc...

![Population Chart]

*Population*

<table>
<thead>
<tr>
<th>Category</th>
<th>Percentage</th>
</tr>
</thead>
<tbody>
<tr>
<td>Radiology</td>
<td>35.00</td>
</tr>
<tr>
<td>Computer Vision</td>
<td>25.00</td>
</tr>
<tr>
<td>Microscopy</td>
<td>20.00</td>
</tr>
<tr>
<td>Other</td>
<td>10.00</td>
</tr>
</tbody>
</table>
Population that don’t use C++

![Percentage of Subjects that don't use C++](chart.png)

Note:
- In this survey, most of the people that don’t use C++ are from microscopy
<table>
<thead>
<tr>
<th></th>
<th>Software / Libraries</th>
<th></th>
<th>Software / Libraries</th>
<th></th>
<th>Software / Libraries</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>3D Slicer</td>
<td>16</td>
<td>Irfanview</td>
<td>34</td>
<td>OpenSceneGraph</td>
</tr>
<tr>
<td>2</td>
<td>Analyze</td>
<td>17</td>
<td>Vxl</td>
<td>35</td>
<td>OTB</td>
</tr>
<tr>
<td>3</td>
<td>Cimage</td>
<td>18</td>
<td>Matlab</td>
<td>36</td>
<td>ParaView</td>
</tr>
<tr>
<td>4</td>
<td>convert3D</td>
<td>19</td>
<td>Octave</td>
<td>37</td>
<td>PIL</td>
</tr>
<tr>
<td>5</td>
<td>MIPAV</td>
<td>20</td>
<td>Qt</td>
<td>38</td>
<td>Mayavi</td>
</tr>
<tr>
<td>6</td>
<td>DCMTK</td>
<td>21</td>
<td>OsiriX</td>
<td>39</td>
<td>R</td>
</tr>
<tr>
<td>7</td>
<td>ENVI</td>
<td>22</td>
<td>Simpleware ScanIP</td>
<td>40</td>
<td>Seg3D</td>
</tr>
<tr>
<td>8</td>
<td>Orfeo Toolbox</td>
<td>23</td>
<td>Mathematica</td>
<td>41</td>
<td>V3D</td>
</tr>
<tr>
<td>9</td>
<td>FSL</td>
<td>24</td>
<td>GPU shaders</td>
<td>42</td>
<td>VisualC++</td>
</tr>
<tr>
<td>10</td>
<td>Camino</td>
<td>25</td>
<td>Mia.sf.net</td>
<td>43</td>
<td>ANTS</td>
</tr>
<tr>
<td>11</td>
<td>ITK-snap</td>
<td>26</td>
<td>Mipav</td>
<td>44</td>
<td>Numpy</td>
</tr>
<tr>
<td>12</td>
<td>GDCM</td>
<td>27</td>
<td>Etdips</td>
<td>45</td>
<td>Scipy</td>
</tr>
<tr>
<td>13</td>
<td>Gimp</td>
<td>28</td>
<td>Mitk</td>
<td>46</td>
<td>ImageMagick</td>
</tr>
<tr>
<td>14</td>
<td>Fiji</td>
<td>29</td>
<td>Vala</td>
<td>47</td>
<td>Clojure</td>
</tr>
<tr>
<td>15</td>
<td>Haskell</td>
<td>30</td>
<td>My own code</td>
<td>48</td>
<td>Javascript</td>
</tr>
<tr>
<td></td>
<td></td>
<td>31</td>
<td>Hugin</td>
<td>49</td>
<td>VTK</td>
</tr>
<tr>
<td></td>
<td></td>
<td>32</td>
<td>IDL</td>
<td>50</td>
<td>ImageJ</td>
</tr>
<tr>
<td></td>
<td></td>
<td>33</td>
<td>In-house Tools</td>
<td>51</td>
<td>OpenCV</td>
</tr>
</tbody>
</table>
What libraries are used to process images (besides ITK)?

Use of Three Open Source Libraries

<table>
<thead>
<tr>
<th>Library</th>
<th>Num. of Users</th>
</tr>
</thead>
<tbody>
<tr>
<td>VTK</td>
<td>140</td>
</tr>
<tr>
<td>ImageJ</td>
<td>100</td>
</tr>
<tr>
<td>OpenCV</td>
<td>50</td>
</tr>
</tbody>
</table>

Use of three Imaging Libraries among Different Background

**Q:**

- Is ImageJ the primary software in microscopy?
Libraries used among non-C++ developers

Among Non-C++ Developers

- ImageJ
- VTK
- Matlab
- OpenCV
- FSL
- ImageMagick

Note that all of the top libraries / applications among non-C++ developers provide processing and visualization.
ITK
Functional vs. Object Oriented

**Example 1: Functional**

```
Image out = GaussianFilter::Blur(inImage, 2);
```

**Example 2: Object-oriented**

```
GaussianFilter filter;
filter.SetSigma(2);
Image out = filter.Blur(inImage);
```
Overall, it seems that people prefer an *Object Oriented* programming style. However....
Functional vs. Object Oriented

Preference Among Non-C++ Programmers

- Functional: 46.51%
- Object Oriented: 27.91%
- No Preference: 25.58%

Preference Among Non-ITK Users

- Functional: 29.73%
- Object Oriented: 37.84%
- No Preference: 32.43%

However:
- Non-C++ programmers prefer a Functional style
- Non-ITK users don’t have strong preferences
How well do you understand the pipeline?

**Filter Pipeline**

Note:
Most non-C++ users (including the 51% that have used ITK), don’t understand the pipeline.
How well do you understand streaming?

![Graph showing understanding of streaming]

**Note:**
Most ITK, non-ITK, and C++ users don’t have a clear understanding of streaming.

**However....**
Loading images larger than RAM?

Overall: Do you need to load images larger than local memory?

- No: 62.68%
- Yes: 46.53%
Loading images larger than RAM?

Non-C++ Users: Do you need to load images larger than local memory?

ITK Users: Do you need to load images larger than local memory?

Microscopy: Do you need to load images larger than local memory?
Programming Languages

The most important programming languages?
Programming Languages

Preferences: C++ Experts

Preferences: Non C++ Developers

Q: Can we say that Matlab, Python, C++, Java, and R are the most important programming languages?
Other programming languages

- Bash
- C
- Fortran
- JavaScript
- Mathematica
- Objective-C
- Octave
- PHP
- D
- Vala
- TCL
- Visual Basic
Simplicity vs. Performance

Performance Among Groups

Order: 4 2 1 3 5

It seems that for non-C++ and non-ITK developers simplicity is more important than performance.
What ITK features are important?

Note:
A **good** and **simple** segmentation technique is better than a complex and state-of-the-art approach?
What ITK features are important?

It seems that the most important features are:
1. Registration
2. Filters
3. Region Growing
4. Convert
Pixel Type: Overall

![Bar chart showing pixel type overall importance across different data types.

- More important data types include 'float', 'uchar', 'ushort', 'short', 'double', 'Matrix', 'int', 'RGB3', 'uint', 'char', 'RGB4', 'ulong', and 'long'.
- Not important data types include 'float', 'uchar', 'ushort', 'short', 'double', 'Matrix', 'int', 'RGB3', 'uint', 'char', 'RGB4', 'ulong', and 'long'.
- Low Priority data types include 'float', 'uchar', 'ushort', 'short', 'double', 'Matrix', 'int', 'RGB3', 'uint', 'char', 'RGB4', 'ulong', and 'long'.
- High Priority data types include 'float', 'uchar', 'ushort', 'short', 'double', 'Matrix', 'int', 'RGB3', 'uint', 'char', 'RGB4', 'ulong', and 'long'.
- Essential data types include 'float', 'uchar', 'ushort', 'short', 'double', 'Matrix', 'int', 'RGB3', 'uint', 'char', 'RGB4', 'ulong', and 'long'.]
Pixel Type: Non-ITK Users

Top 6 (Non-ITK users)
1. uchar
2. double
3. ushort
4. float
5. RGB3
6. short

Top 6 (Overall)
1. float
2. uchar
3. ushort
4. short
5. double
6. matrix
Pixel Type: Non-ITK Users

- Binary / bool images
- Imagej2
- Fiji
- Bool
- Tensor
- Multichannel
- NIFTI
Platform

• Other:
  – Cluster / grids
  – Phone / mobile apps
  – GPU-based cluster
  – iPad
  – Services on servers
Comments: Default Parameters

1. “Important to give users relatively straightforward advice on algorithm...”

2. “Simple defaults with optional alternatives”

3. “I generally want to be able to try something off the shelf with sensible defaults”

4. “Sensible defaults should be provided to avoid long code”

5. “Full flexibility is needed but a good selection of default parameters for all modules would be also important”

6. “initialized with reasonable default components”
Comments:  Display Images

• “offer both...a GUI ... and build scripts from there”

• “connect display to view intermediate images”

• “handling ability to visualize quickly (even at run time) images and histograms”

• “A GUI to build up the application pipeline by arranging icons of filters and I/O connectors”

• “visual tool for connecting filters to a pipeline
Comments: Documentation

• “writing ITK codes can be a complex task but with more examples and thorough documentation these difficulties can be overcame”

• “One good way to compromise is more flexible but with lots of example code that can be cut-and-pasted for typical applications.”

• “Actually I think much of the motivation for SimpleITK could go away, if someone updated + improved the documentation”

• “some of the image filters modules are not sufficiently documented”

• “Lots of sample code that can be modified for specific purposes”
Comments: Other

• Features:
  – Graph-cut (e.g. boost graph-cut lib)
  – Tracking
  – Distributed I/O
  – Fast morphological operators

• General comments:
  – “It's better to provide fewer full-featured classes than to provide many partially-featured classes”
  – “I am not using ITK for registration purposes, and this is because of the complexity of it”
  – “Extensive use of templates makes debugging difficult in VisualC++”
  – “Most of filter called *VectorImageFilter are in fact not able to process itk::VectorImage ... This is confusing and sometimes frustrating”

• Frustrated folks:
  – Actually, ITK registration framework is not modular enough to be useful.
  – Ability to open differing file formats (dimensions, data types) without all the stupid c++ template nonsense.