Bildverarbeitung und Algorithmen
Prof. Dr. Wolfgang Konen

Einführung in
ImageJ

Tools in der Bildverarbeitung

- Früher:
  - fast so viele BV-Tools wie BV-Formate
  - Lösungen nur schwer auf andere Systeme übertragbar
- Der Grund: Performance
  - BV = viele Pixel = hohe Rechenleistung
- Heute: Extensibility/Plattformunabhängigkeit im Vordergrund

Was ist ImageJ? (1)

- Open-Source-Projekt, initiiert durch Wayne Rusband, NIH
- Philosophie:
  - kompakter, aber dennoch mächtiger Kern von Basisoperationen
    - Bilder einlesen / schreiben
    - Bilder **be**arbeiten
    - Bildverbesserung
    - Bilder analysieren
  - einfach erweiterbar durch Java-Plugins, die on-the-fly erstellt und dem System hinzugefügt werden können
  - inzwischen große User-Gemeinde, darum große Zahl an Plugins
  - http://rsb.info.nih.gov/ij/
  - http://ij-plugins.sourceforge.net/
  - Erste ImageJ-Konferenz, 18-19.05.06, Luxemburg: http://imagejconf.tudor.lu/

ImageJ – Allgemeine Eigenschaften

- **Runs Everywhere:**
  - ImageJ läuft auf
    - Linux,
    - Mac OS 9, Mac OS X,
    - Windows,
    - und Sharp Zaurus PDA.
- **Open Source:**
  - ImageJ and its Java source code are freely available and in the public domain. No license is required.
- **User Community:**
  - ImageJ has a large and knowledgeable worldwide user community. More than 1200 users and developers subscribe to the ImageJ mailing list.

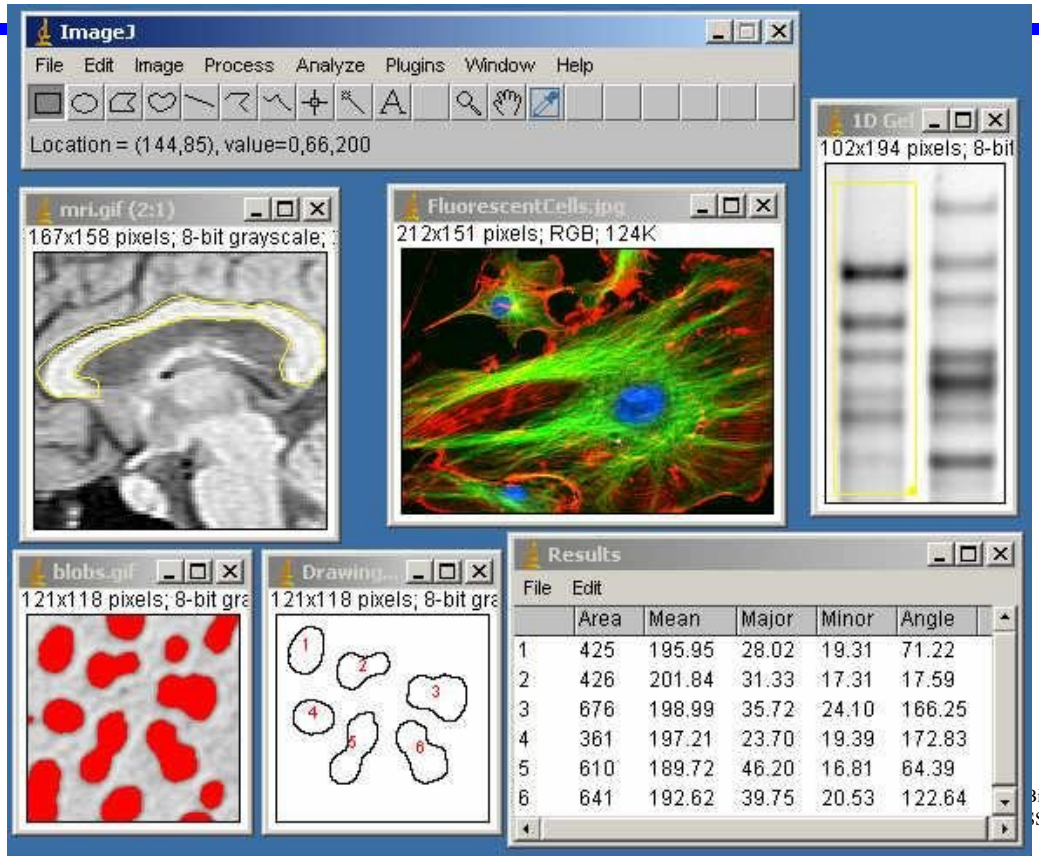
Wie kann man ImageJ einsetzen?

auf 4 Ebenen

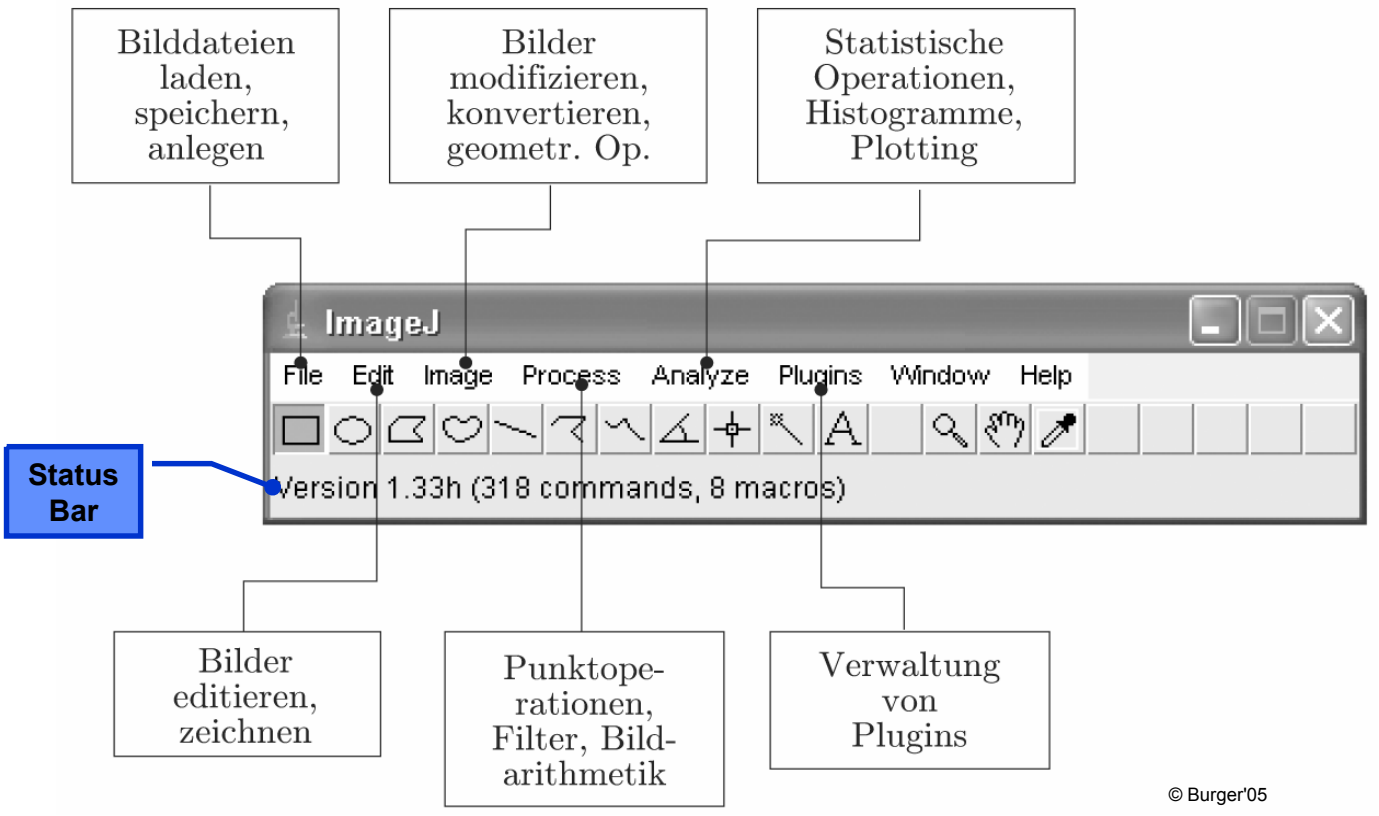
1. ImageJ als Toolbox nutzen (manuelle Bildbearbeitung und Analyse)
2. Macros aufzeichnen oder schreiben
   - komplexere Abläufe automatisieren
3. ImageJ-Plugins schreiben
   - erfordert Java-Kenntnisse
   - Rapid Prototyping >> schnelle Integration
4. ImageJ in eigene Applikation einbinden
   - auch als Applet oder Servlet möglich

Wir nutzen vor allem 1. und 3., manchmal auch 2.

Basiskonzepte ImageJ

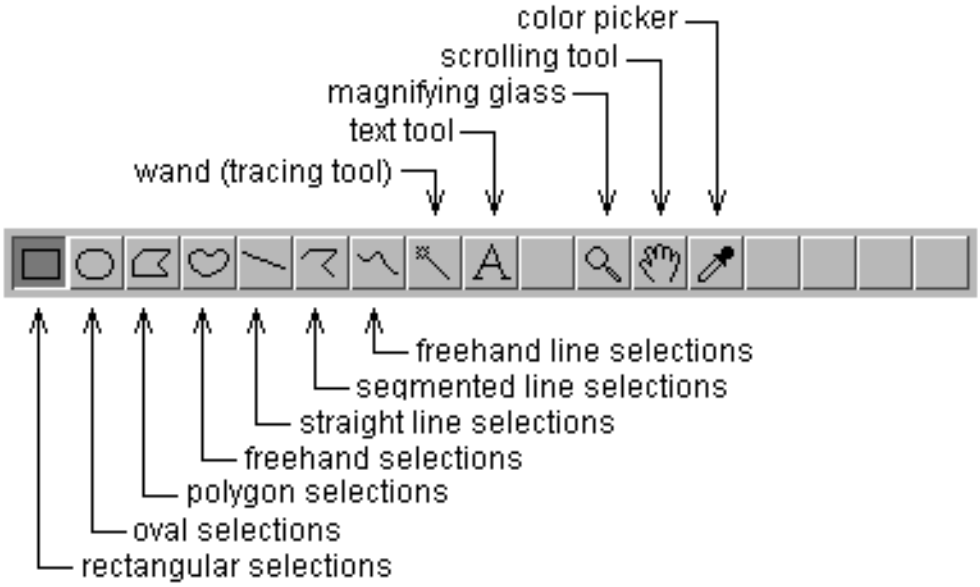


Menus



© Burger'05

Tools

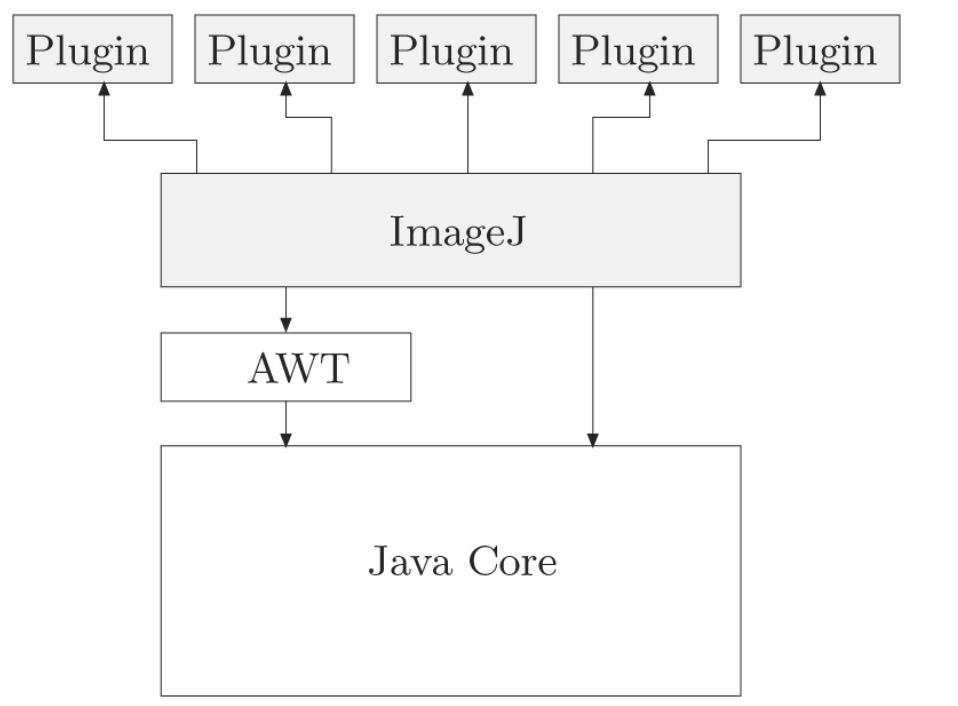


Software Demonstration

Walk Through
ImageJ +
Documentation

- ... dann Übungen Ü1-Ü4

Software-Konzept



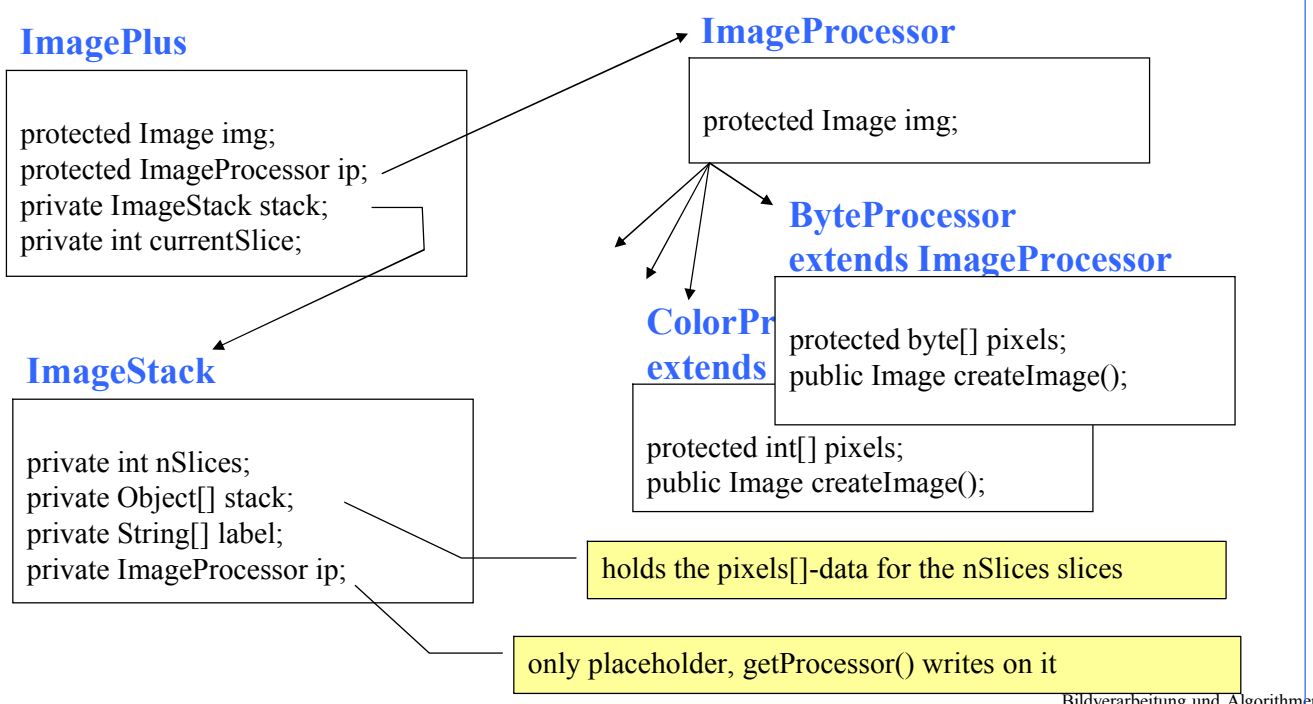
© Burger'05

Einführung Plugin-Programmierung

Die Einführung erfolgt anhand der Handouts [Burger05] u. [Bailer01] sowie praktischer Übungen, und zwar nach folgendem Fahrplan:

- ☐ Walk-Through S. 31-35 aus [Burger05]: **Erstes Plugin-Beispiel**
- ☐ ... selber ausprobieren am Rechner
- ☐ Walk-Through S. 4-8 aus [Bailer01]: **ImageJ Class Structure**
- ☐ Walk-Through S. 12-16 aus [Bailer01]: **Image Representation, ROI**
- ☐ ... dann Übungen Ü5-Ü6
- ☐ [Bailer01, S. 9-10]: **Inverter_-Beispiel erweitert: ROI, schneller**
- ☐ ... dann Übungen Ü7-Ü9

Zusammenspiel der Klassen ImagePlus, ImageProcessor, ImageStack



Nützliche weitere Hinweise

- Mit Plugins – New – PluginFilter... bekommt man direkt ein Template mit den richtigen `imports` und Methoden
- Ein `Rectangle` ist eine Klasse mit den public Membern
  - x,y: obere linke Ecke
  - width,height
- Mit `Rectangle ip.getRoi()` bekommt man das Bounding Rectangle einer ROI im `ImageProcessor ip` zurück
- ACHTUNG: `ROI implus.getRoi()` aus der Klasse `ImagePlus` ist eine andere Methode (!!)
- Der Plot erscheint in einem `PlotWindow` [Bailer01, 6.6.3] erst, nachdem dessen Methode `draw()` aufgerufen wurde (!)