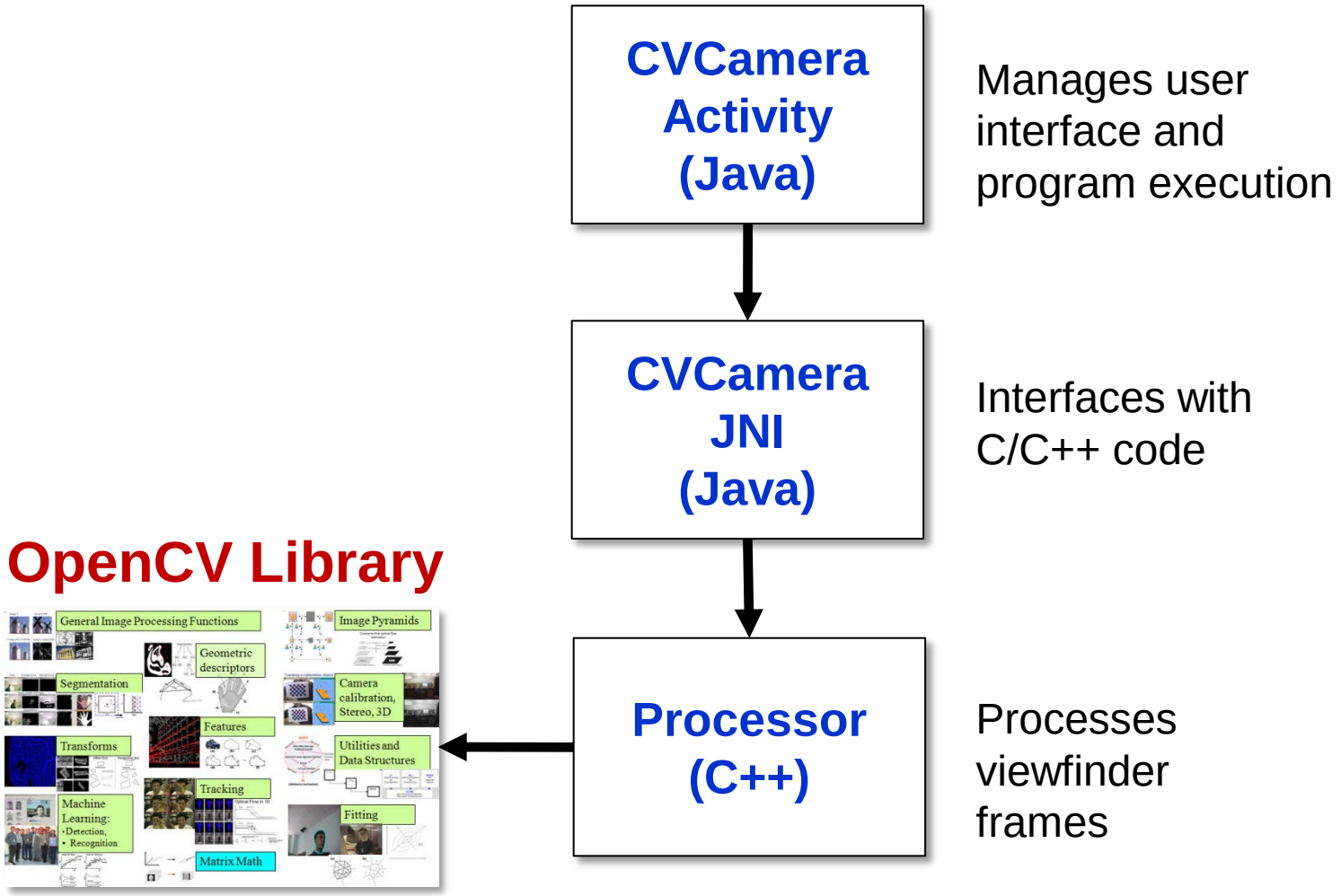
“CVCamera” project

- Goals of this project
  - Learn how to incorporate C/C++ code into an Android project
  - Learn how to utilize OpenCV library functions
  - Learn how to draw feature keypoints on viewfinder frames

- Full source available on class website

“CVCamera” class hierarchy



CVCamera class: menu options

```
public boolean onCreateOptionsMenu(Menu menu) {
    menu.add("FAST");
    menu.add("STAR");
    menu.add("SURF");
    menu.add("MSER");
    menu.add("Settings");
    return true;
}
```

Java Code

CVCamera class: calling feature extractors

```
public boolean onOptionsItemSelected(MenuItem item) {
  LinkedList<PoolCallback> defaultCallbackStack =
       new LinkedList<PoolCallback>();

  defaultCallbackStack.addFirst(myGLView.getDrawCallback());

  if (item.getTitle().equals("FAST")) {
       defaultCallbackStack.addFirst(new FastProcessor());
  }
  else if (item.getTitle().equals("STAR")) {
       defaultCallbackStack.addFirst(new STARProcessor());
  }
  else if (item.getTitle().equals("SURF")) {
       defaultCallbackStack.addFirst(new SURFProcessor());
  }
  else if (item.getTitle().equals("MSER")) {
       defaultCallbackStack.addFirst(new MSERProcessor());
  }

  myPreview.addCallbackStack(defaultCallbackStack);
  return true;
}
```

Java Code

Populate a stack of callback functions

First callback function draws the frames

Other callback functions call feature extractors

“CVCamera” class hierarchy

CVCamera Activity (Java)

Manages user interface and program execution

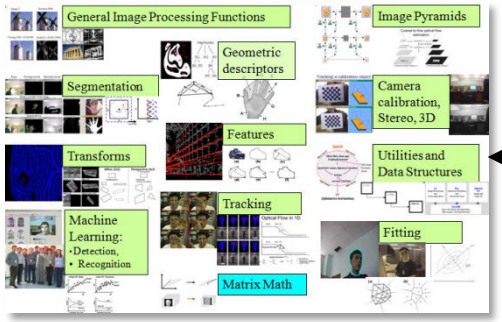
CVCamera JNI (Java)

Interfaces with C/C++ code

Processor (C++)

Processes viewfinder frames

OpenCV Library



JNI class: interface to C/C++ code

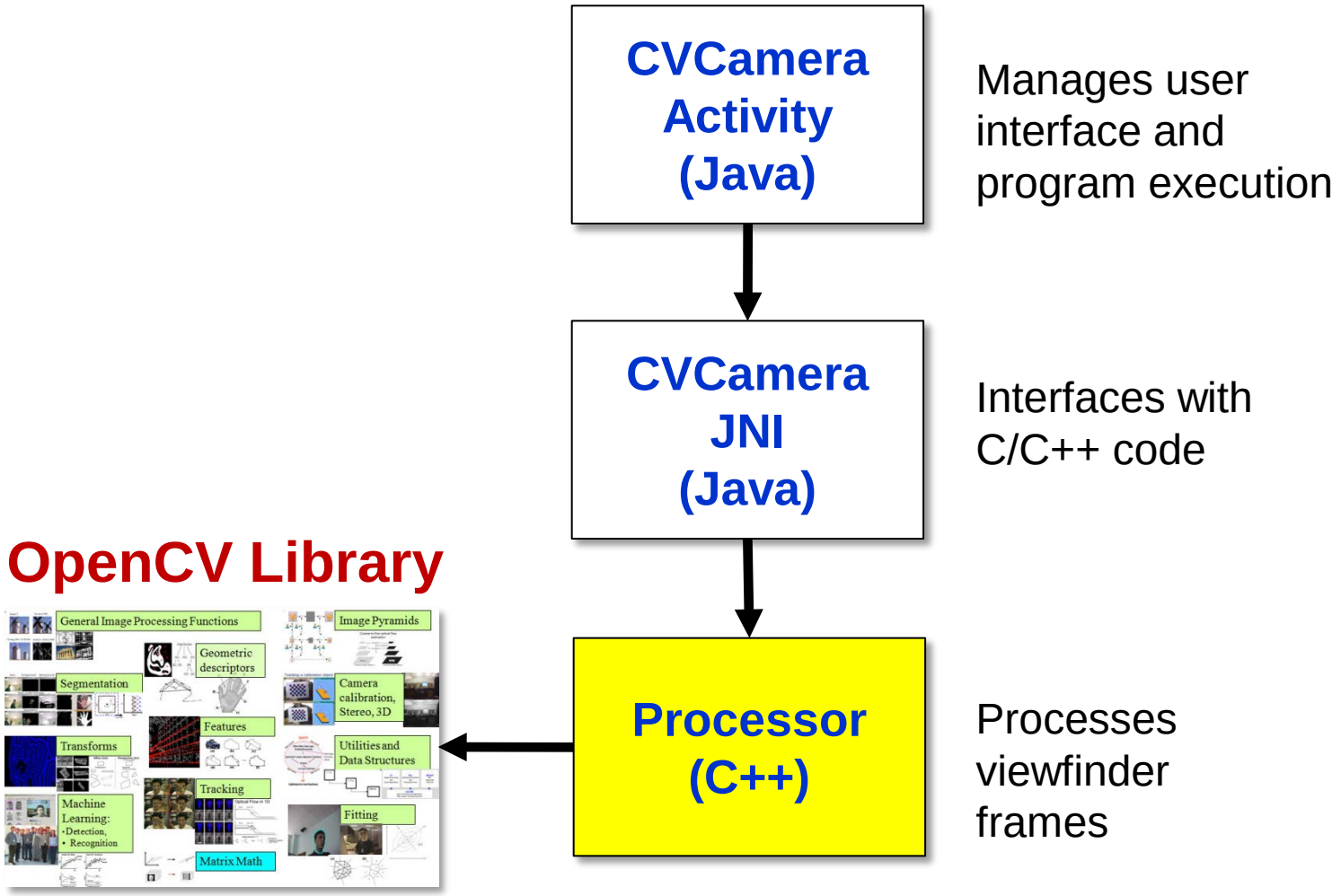
```
static {
  try {
        System.loadLibrary("android-opencv");
        System.loadLibrary("cvcamera");
  } catch (UnsatisfiedLinkError e) {
        throw e;
  }
}

public final static native int DETECT_FAST_get();
public final static native int DETECT_STAR_get();
public final static native int DETECT_SURF_get();
public final static native int DETECT_MSER_get();
public final static native long new_Processor();
public final static native void delete_Processor(long jarg1);
```

Java Code

Load libraries built by the Android NDK

Function prototypes for interface to C/C++ side

“CVCamera” class hierarchy



Processor class: initialize feature detectors

```
class Processor {
private:
  cv::StarFeatureDetector my_stard;
  cv::FastFeatureDetector my_fastd;
  cv::SurfFeatureDetector my_surfd;
  cv::MserFeatureDetector my_mserd;
  std::vector<cv::KeyPoint> my_keypoints;

public:
  Processor():
    my_stard(STAR detector parameters),
    my_fastd(FAST detector parameters),
    my_surfd(SURF detector parameters),
    my_mserd(MSER detector parameters)
    {}
  void detectAndDrawFeatures
    (int idx, image_pool* pool, int feature_type);
};
```

C++ Code

Different feature extractors stored as members

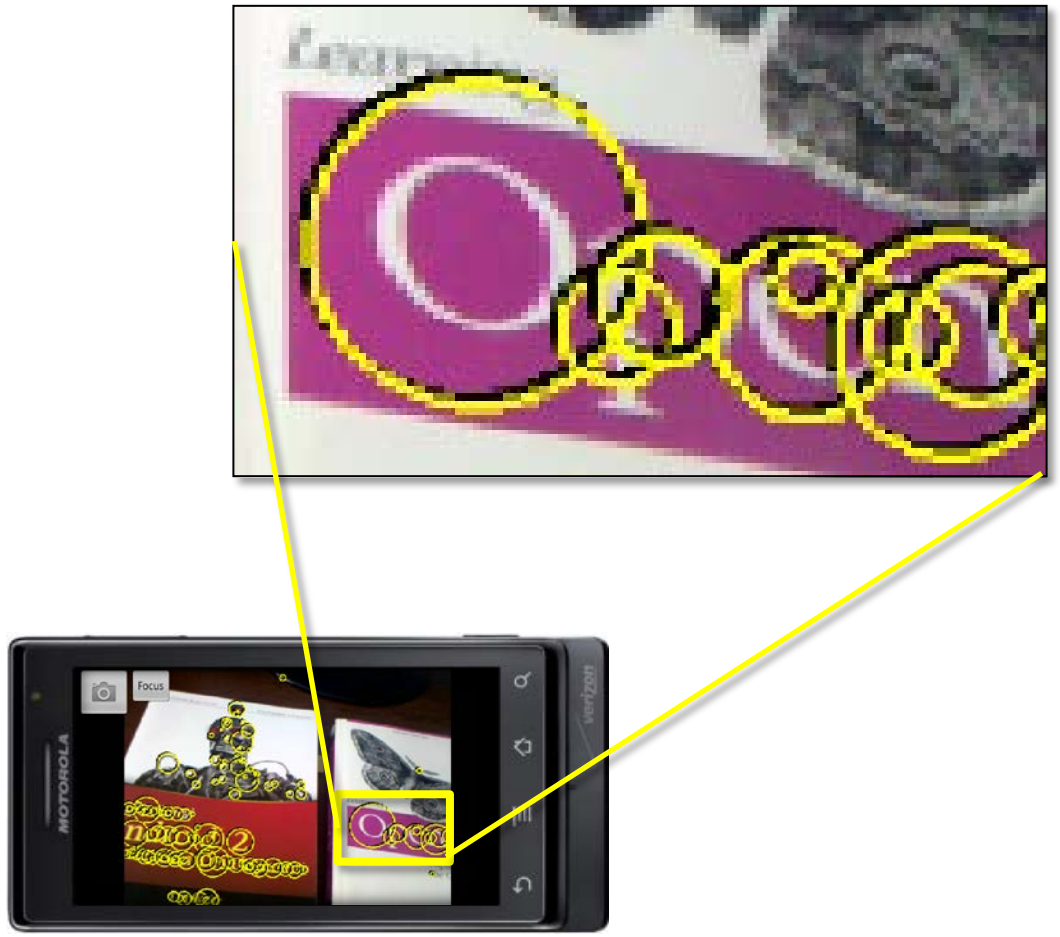
Initialize extractors in the class instantiation list

Function for detecting and drawing keypoints

Processor class: detect and draw feature keypoints

```
// Detect feature keypoints
Mat grey_im = pool->getGrey(idx);
Mat color_im = pool->getImage(idx);
my_keypoints.clear();
my_mserd->detect(grey_im, my_keypoints);

// Draw feature keypoints
vector<KeyPoint>::const_iterator it;
for (it = my_keypoints.begin();
     it != my_keypoints.end(); ++it) {

  // Draw black circle
  Point2f pt = it->pt;
  circle(color_im, pt, it->size,
         cvScalar(0,0,0,0), 2);

  // Draw yellow circle
  pt.x += 1;  pt.y += 1;
  circle(color_im, pt, it->size,
         cvScalar(255,255,0,0), 2);
} // iterator
```

C++ Code



Local feature keypoints extraction