CVBPL
TEAM 4

Team Members

Brainstorming

Features

Language Design

Language Design

Technology We Used

Test Plan

Architecture

Demonstration

More demos

Lessons Learned

Thank You!

CVBPL

TEAM 4

Team Members

Rui Cao (System Integrator)

Qiu Feng Chen (Verification and Validation)

Jiawen Sun (Project Manager)

Ioan Daniel Negulescu (Language Guru)

Sida Yang (System Architect)

Brainstorming

Interested in image processing?

Hate C and Java's complex functions?
Matrix m = m1.add(m2.sub(m3).div(m4))?

Is there a better way for us to manipulate images?

What do we really need?

Brainsto

Interested in image processing?

Hate C and

e processing?

Hate C and Java's complex functions?
Matrix m = m1.add(m2.sub(m3).div(m4))?

way for us to
es?

What do we really need?

Hate C a
Matrix m

Is there a better way for us to manipulate images?


).div(m4))?
What do we really need?

Why CVBPL?

- Domain-oriented
- Simple and Clean
- Powerful

Applications

- Gaussian blurring
- Edge detection
- Filters and transformations
- Noise reduction
- Object identification

Outline

- Language Design
- Compiler Implementation
- Implementation Testing
- Demonstration
- Lessons and Conclusion

```
Func: {
    int fib (int i) {
        if (i == 0) return 1;
        else if (i == 1) return 1;
        else return fib (i - 1) + fib (i - 2);
    }
}

Prog: {
    int i = 0;
    while(i < 10) {
        printInt (fib (i) );
        i = i + 1;
    }
}
```

Syntactic Constructs

Translation unit structure:

- **Func Block - function declarations**
- **Prog Block - entry point and control**

Control flow:

- **if-else statement**
- **while statement**

Variable and function declaration

- **Language is statically and strongly typed**

Compound statements introduce scope

Types:

- **int, float ("numeric types")**
- **String, Image, Matrix, Histogram, Handle ("complex types")**

```
=== Method: fib ===
Code(max_stack = 4, max_locals = 1, code_length
= 66)
0:   iconst_0
1:   iload_0
2:   ldc  0 (7)
4:   if_icmpeq  #11
7:   iconst_0
8:   goto  #12
11:  iconst_1
12:  nop
13:  dup
14:  ifeq  #20
17:  ldc  1 (8)
19:  ireturn
20:  nop
21:  ifne  #64
24:  iload_0
25:  ldc  1 (8)
27:  if_icmpeq  #34
. . .
```

```
===
Co
co
0:
2:
3:
4:
5:
7:
10:
11:
14:
15:
16:
19:
20:
. . .
```

```
=== Method: main ===
Code(max_stack = 2, max_locals = 2,
code_length = 40)
0:   ldc  0 (7)
2:   istore_1
3:   nop
4:   iload_1
5:   ldc  10 (17)
7:   if_icmplt  #14
10:  iconst_0
11:  goto  #15
14:  iconst_1
15:  nop
16:  ifeq  #38
19:  iload_1
20:  invokestatic fib.fib (I)I (11)
. . .
```

Complex Types

- Represent domain entities (e.g. Matrix, Image, Histogram) and supporting types (e.g. Handle)

- May have properties (l- or r-valued)
  - e.g.: int h = matrix.height;
  - r-valued properties may not be assigned to

```
Error in program: height / width properties are not lvalues
```

- May have explicit constructors:
  - e.g.: Matrix m(height, width);

- Passed by reference

- Type system properties
  - statically and strongly typed
  - all type errors detectable at compile time

```
Error in program: Invalid operation: num / Matrix
```

  - automatic int-to-float type promotion

Complex Types

Image: represents a 3-channel RGB image

- **Image (int height, int width)**
- **int height, width**
- **Matrix red, green, blue [lvalues]**

Matrix: represents a 2-dimensional matrix

- **Matrix (int height, int width)**
- **int height, width**

Histogram: provides count and statistics of colors in a Matrix

- **Histogram (Matrix m)**
- **float mean, median, mode, min, max**
- **int count**

Handle: provides read / write access to a file

- **Handle (String file, String mode)**
  - **mode is "r" or "w"**

Rich Set of Operators and Constructs

Arithmetic: + - * /

- Statements such as Matrix + Numeric perform element-wise operations
- Matrix-Matrix (when valid)
- Matrix-Numeric
- Numeric-Matrix (when valid)
- Numeric-Numeric

Convolution: **

- Image-Matrix
- Matrix-Matrix

Transpose: '

- Matrix

Relational + Logical: < <= >= > == != !

Unary: -

I / O:

- Image << String
- Image >> String

Submatrix construction:

- Matrix m(r1:r2; c1:c2)
- All expressions in the range are optional; inferred if missing

Matrix / Histogram access:

- float m[row, col]
- int h[intensity]

Standard Library

Math
- float sqrt (float f)
- float sin (float f)
- float cos (float f)
- float tan (float f)
- float atan (float f)
- float atan2 (float f1, float f)

I/O
- void print(String s)
- void println(String s)
- void printInt(int i)
- void printFloat(float f)
- Handle getSTDIN()
- Handle getSTDOUT()
- int readInt(Handle h)
- float readFloat(Handle h)
- String readString(Handle h)
- String readLine(Handle h)
- void writeInt(Handle h, int i)
- void writeFloat(Handle h, float f)
- void writeString(Handle h, String s)
- void writeLine(Handle h, String s)
- void close()

Conversion
- int ftoi (float f)
- int stoi (String s)
- String itos (int i)
- String ftos (float f)
- float stof (String s)

```
Prog: {
   Image input;
   input << "test.png";

   Image output(input.height, input.width);

    int i = 2, j = 2;
    while(i < (input.height - 1)) {
       while(j < (input.width - 1)) {
          Matrix regionR = input.red(i-1:i+1; j-1:j+1);
          Matrix regionG = input.green(i-1:i+1; j-1:j+1);
          Matrix regionB = input.blue(i-1:i+1; j-1:j+1);
          Histogram hR (regionR), hG (regionG), hB (regionB);

          float medRed = hR.median, medBlue = hB.median, medGreen = hG.median;
          output.red[i, j] = medRed;
          output.green[i, j] = medGreen;
          output.blue[i, j] = medBlue;

          j = j + 1;
       }
       j = 0;
       i = i + 1;
   }
   output >> "mediantest.png";
}
```

Remarks on Bytecode

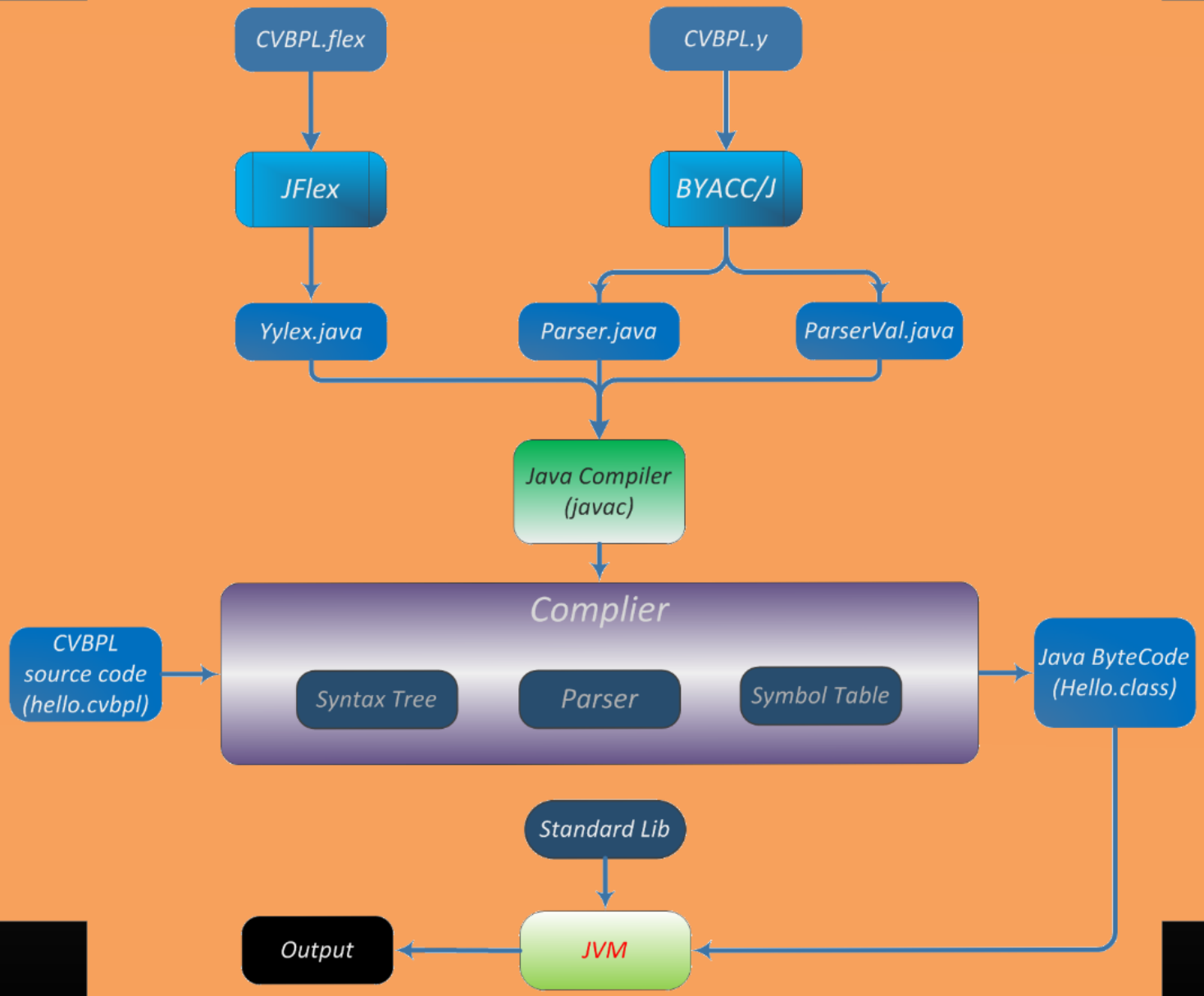
Compilation to bytecode has several benefits:

- Portable - run anywhere there is Java
  - (i.e., everywhere)
- Just in time compiler optimizes our output
- Bytecode produced by our compiler from CVBPL source can be invoked from Java programs.

Technology We Used

Tools & Software	Purpose
JFlex and BYACC/J	Lexical and syntactical analysis
Eclipse IDE	IDE to run and debugging the code
github	Version control
Java	Compiler and library implementation
BCEL	Generate byte code
Gmail, Google Group and Google Docs	Team communication
Unfuddle	Bug tracker

Architecture



Unit Testing

Integration Testing

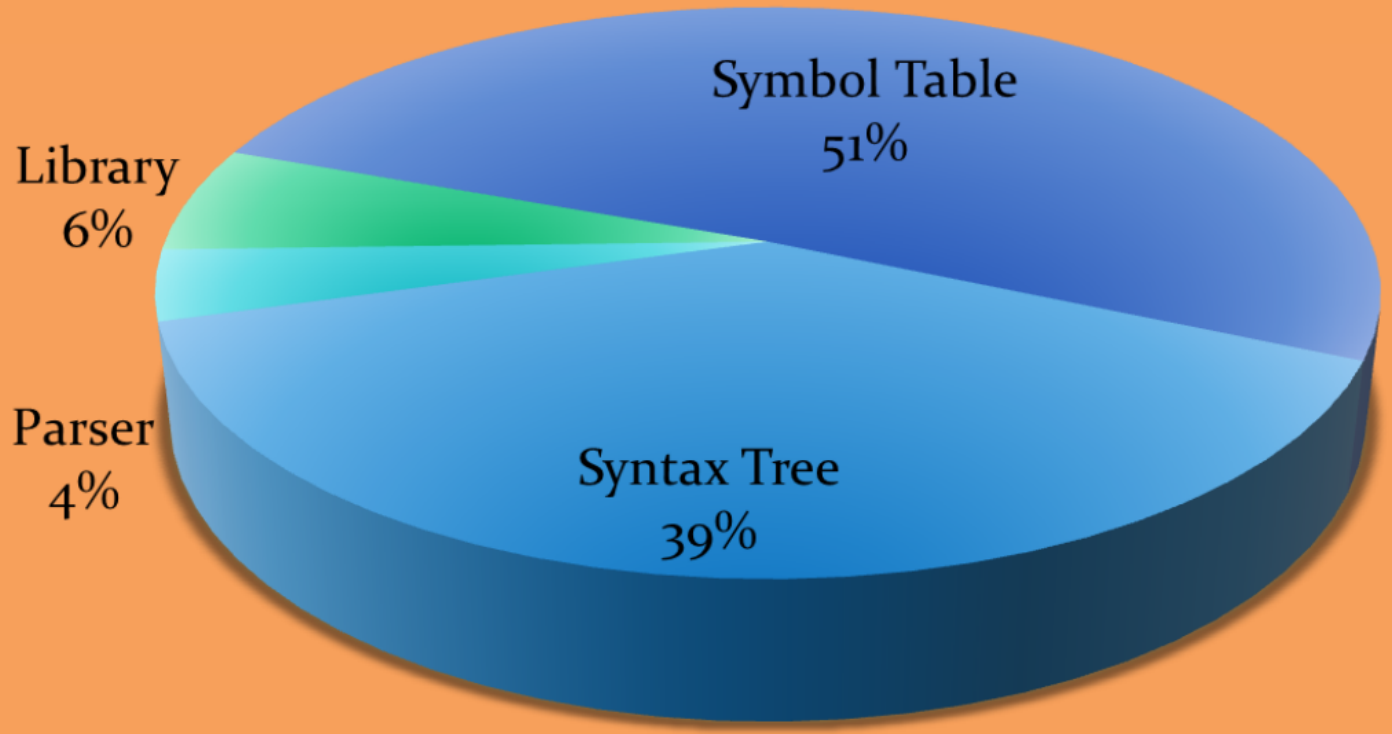
Verification and Validation

Regression Testing

Processing Testing

Test Plan

Error Densities



Unit

Integration

Veri

This program converts an color image to a grayscale image. It shows our convenient input and output operators. Complex type arguments are passed by reference.

```
Func: {
    void rgbToGray(Image input, Image output) {
        Matrix gray = 0.2989 * input.red
            + 0.5870 * input.green + 0.1140 *
            * input.blue;
        output.red = output.green = output.blue =
            gray;
    }
}
Prog: {
    Image input, output;
    input << "./INPUT/lena.bmp";
    output << "./INPUT/lena.bmp";
    rgbToGray(input, output);
    output >> "./OUTPUT/grayLena.bmp";
}
```

This program first converts an color image into a grayscale image, then convolves the grayscale image with an isotropic edge detector. The output is an edge image. It shows our matrix access and convolution operator.

```
Image input, output, storage1, storage2;
Matrix matIN, matOUT;
storage1 << "./INPUT/lena.bmp";
matIN = storage1.red;
storage2 << "./INPUT/lena.bmp";
matOUT = storage2.red;
input << "./INPUT/lena.bmp";
rgbToGray(input,matIN);
Matrix edgeMask(3,3);
edgeMask[0,0] = 0.125;
edgeMask[0,2] = 0.125;
edgeMask[2,0] = 0.125;
edgeMask[2,2] = 0.125;
edgeMask[0,1] = -0.25;
edgeMask[1,0] = -0.25;
edgeMask[1,2] = -0.25;
edgeMask[2,1] = -0.25;
edgeMask[1,1] = 0.5;
matOUT = matIN ** edgeMask;
```

More demos

MedianDenoising.cvbpl


Sobel1.cvbpl & Sobel2.cvbpl



GenerateHue.cvbpl


MedianFilter.cvbpl


Scenery 1

Scenery 2

Scenery 3

Test 1

Test 2

Test 3

```
ruicao@ubuntu:~/workspace/CVBPL/bin$ cv ObjectClassify
Object/Scenery Classification!
Feature for Scenery 1:  0.0 4.0 58.0 78.0 159.0 325.0 16.0 0.0 0.0 0.0
Feature for Scenery 2:  572.0 8.0 0.0 4.0 29.0 13.0 5.0 4.0 0.0 5.0
Feature for Scenery 3:  4.0 33.0 177.0 75.0 43.0 114.0 190.0 2.0 0.0 2.0
Feature for Test Scenery:  568.0 24.0 1.0 0.0 4.0 8.0 14.0 4.0 4.0 13.0
Distance between test scenery and scenery 1:  457076
Distance between test scenery and scenery 2:  1100
Distance between test scenery and scenery 3:  398652
test scenery should be the same as scenery 2.
actual test scenery is ./INPUT/test2.jpg
```

Lessons Learned

- *Automate regression testing early*
- *Understand the big picture - how does each part connect to the others?*
- *Eliminate process inconveniences*
- *Give each member a clear task at the end of every meeting*

Thank You!