- PICEL -

PICture Editing Language

Manager | Chia-Hao Hsu | ch3141

System Architect | Chih-Sheng Wang | cw2952

Language Guru |Ruijie Zhang | rz2337

Language Guru | Chang Liu | cl3418

Testing | Rui Lu | rl2784

INTRODUCTION & MOTIVATION

In recent days, there are more and more images existing online. Among photo-sharing sites, Facebook had about 20 billion as of last year and Yahoo!-owned Flickr owned over 3.4 billion. Meanwhile, users are eager to make fancier effect on their picture. However, there is no existing programming language specifically designed for manipulating pictures, the software developers still need to use some inappropriate programming language to create the picture editing software. This problem gave us an idea: Why cannot we create a programming language specific for picture editing? Therefore, here comes our new language --- **PICEL**. We would like to create a programming language to support pixel manipulation and picture editing operation in programming language.

Description

PICEL is the programming language to make programmer edit picture easier. And to make the language speed fast, PICEL is designed to be translated into LLVM IR. PICEL simplify the process of picture editing process by allowing programmers to edit picture and code their program at the same time. PICEL defines several builtin data type to support picture editing, like pixel, picture, array. Also, there are many handy built-in graphic function in this language, like resize, save, load, setHue, setBrightness, etc. These built-in function and data type will help users to manipulate picture with more flexibility. For instance, users can format a mosaic picture with simple for loop, setBrightness, and setHue to format a group of pictures with few lines of codes.

WHY LLVM IR?

- LLVM IR is target-independent, which means our compiler supports multiple architecture such as ARM and x86 or even any platform that supports LLVM interpreter.
- We can leverage the state-of-the-art techniques of code optimization in LLVM.
- There are LLVM IR automatic testing tools (e.g. bugpoint) which can benefit our development process.

LANGUAGE & SYNTAX

The following are the detail of our language. There will be several parts:

- Basic Data Type: Some built-in datatype introduction, here we introduce some data type we define for user to build basic program.
- Operator: Some mathematical & boolean operator
- Function Definition: How do we define function in PICEL language
- Control Flow: Loop control, like for & while. And we also support if / elif / else control flow
- Built-in Function: Some function users might feel useful when they do the basic programming operation.
- Graphic Function Library: In this part, we support several basic picture pixel operation in the library.

- Basic Data Type:

Basic Data Type	Description
int	numerical data type
float	numerical data type
bool	logical data type
char	character data type

Composite Data Type	Description	Syntax

array	array of metadata	char s[10];
		s[0]='a';

picture	data structure of a picture. For a n*m picture, we could store it as a vector of	picture a=load("*.bmp");
		pixel p=a[1][1];

	integer with length=n*m*3. Every 3 integer is the HSV of a pixel.	
pixel	the HSV data of a pixel	pixel p; /* p.h p.s p.v */

- Operator:

Operator	Data Type	Description
+-*/	Int, Float	mathematical operations
and or	Bool	logical “and” and “or”
=	From any type to the same type, or Int to Float	assignment
== != < >	Float, Int	comparison of numerical data type

- Function definition:

```
def <return type> <function name>(<parameter list>){
        <function content>
}
```

- Control Flow:

Keyword	Syntax	Description
if, elif, else	If expr{statement}	expr must be bool type
for	For expr do{loop}	expr must be bool type

while	While expr{loop}	expr must be bool type

- Built-In Function:

Function	Input Data Type	Output Data Type	Description
printf	various	int	print the output
scanf	various	int	read the input
heightOf	picture	int	return the height of given picture (0 for empty picture)
widthOf	picture	int	return the width of given picture (0 for empty picture)
save	picture, char	void	save the current picture
load	char	picture	load picture
lengthOf	array	int	return the length of given array (0 for empty picture)

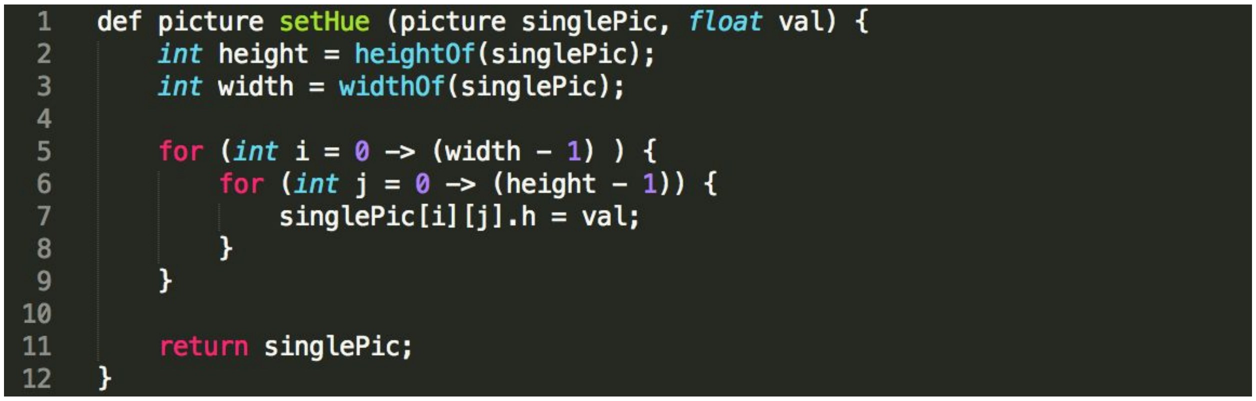
- Graphic Function Library:

Function	Input Data Type	Output Data Type	Description
resize	picture, int, int	picture	resize the picture to the given length and width
setValue	picture, float	picture	set brightness of the picture with the given rate
setHue	picture, float	picture	set hue for the picture with the given rate

setSaturation	picture, float	picture	set saturation for the picture with the given rate

SAMPLE CODE

setHue: In this function, we demonstrate how PICEL help user to adjust the each pixel hue in the picture with two simple for loops:

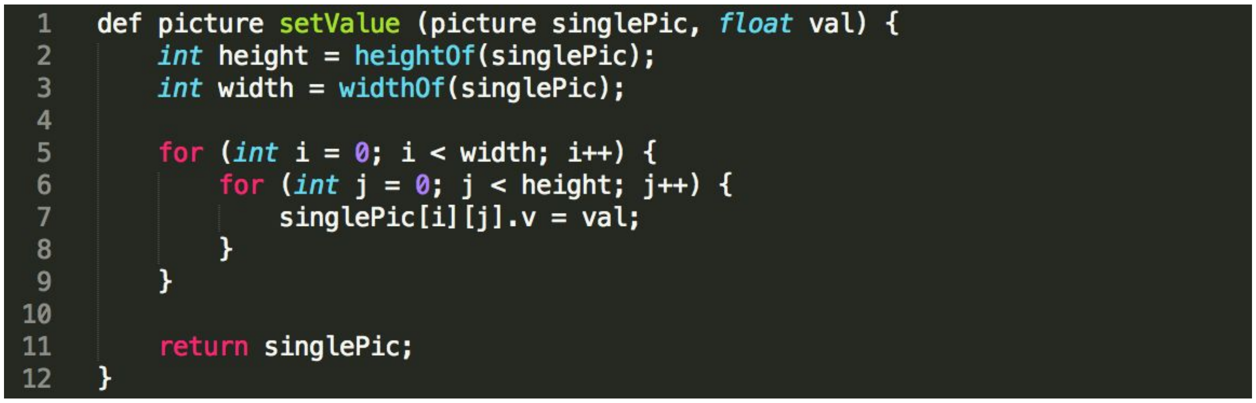


```
def picture setHue (picture singlePic, float val) {
    int height = heightOf(singlePic);
    int width = widthOf(singlePic);

    for (int i = 0 -> (width - 1) ) {
        for (int j = 0 -> (height - 1)) {
            singlePic[i][j].h = val;
        }
    }

    return singlePic;
}
```

setVale: Same as above, here we demonstrate how PICEL help user to adjust the each pixel value in the picture with two simple for loops:

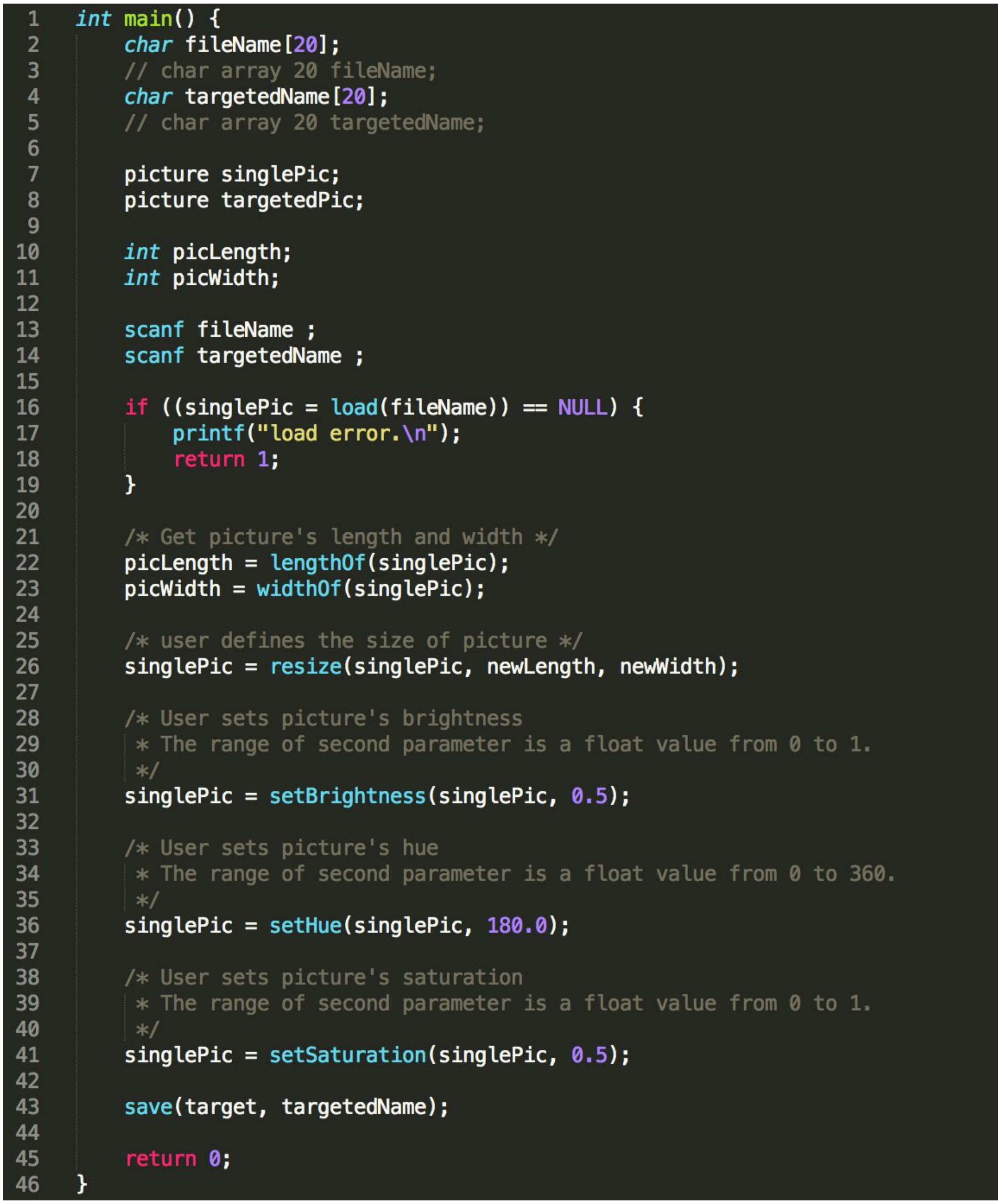


```
def picture setValue (picture singlePic, float val) {
    int height = heightOf(singlePic);
    int width = widthOf(singlePic);

    for (int i = 0; i < width; i++) {
        for (int j = 0; j < height; j++) {
            singlePic[i][j].v = val;
        }
    }

    return singlePic;
}
```

In this main function, we demonstrate how PICEL support user to edit a single picture with several easy built-in graphic function: First, we use picture datatype to store the picture we are about to edit and the new picture we are about to create. And we use resize, setBrightness, setHue, and setSaturation to show how easy it will be to edit a single picture in PICEL



```c
int main() {
    char fileName[20];
    // char array 20 fileName;
    char targetedName[20];
    // char array 20 targetedName;

    picture singlePic;
    picture targetedPic;

    int picLength;
    int picWidth;

    scanf fileName ;
    scanf targetedName ;

    if ((singlePic = load(fileName)) == NULL) {
        printf("load error.\n");
        return 1;
    }

    /* Get picture's length and width */
    picLength = lengthOf(singlePic);
    picWidth = widthOf(singlePic);

    /* user defines the size of picture */
    singlePic = resize(singlePic, newLength, newWidth);

    /* User sets picture's brightness
     * The range of second parameter is a float value from 0 to 1.
     */
    singlePic = setBrightness(singlePic, 0.5);

    /* User sets picture's hue
     * The range of second parameter is a float value from 0 to 360.
     */
    singlePic = setHue(singlePic, 180.0);

    /* User sets picture's saturation
     * The range of second parameter is a float value from 0 to 1.
     */
    singlePic = setSaturation(singlePic, 0.5);

    save(target, targetedName);

    return 0;
}
```