Project Proposal

Fall 2006 PLT
Jianning.Yue (neil.jianning@gmail.com)
Wookyun.Kho (mongdlove@gmail.com)
Young Jin.Yoon (yy2223@columbia.edu)

1. Language Name : IPL(Image Processing Language)

2. Description of the language

- IPL is a functional programming language that enables users to easily describe the picture and animation conceived in his/her mind, and create those images by using the functions such as image displaying, image rotation, color modification, or repeat of images.
- Users easily can make simple animations such as movement of images from location A to B, or connect a series of existing frames to a continuous animation.
- By using IPL, users can define (or import) images he/she likes, and with these images, users can display and modify their own pictures or animation according to personal preference.

3. Features of IPL

- The Image Processing Language (IPL) is designed to facilitate the pattern design and animation production for amateurs who have little professional knowledge in these fields. Image Processing Language (IPL) is a straightforward, productive, interesting, strong, robust and product-transportable language.

- **Straightforward**
- We want to build a complier system that could be programmed easily without much programming experience. It is very easy to use but programmers need to have a picture or animation in mind and trying to think out how to describe it with the language. One has experience in other languages will find the language itself an easy job.

- **Interesting**
- We try to eliminate the tedious elements of common programming language and complex syntax constraints for high flexibility. Besides making the language easy to use, we try to make IPL always arouse programmers' interests in image designing. It is not all about how language works but user will need to be prepared to think very carefully about what will happen when you write different expressions in the language of pictures.

- **Productive**
- IPL is a productive language and flexible on changes of user demand. Programmer can make animations with few prepared materials. Animation producing can be accomplished with a single picture for higher flexibility or a series of frames for higher quality.

- **Strong**
- IPL includes a large resource library that can be employed for pattern design and animation generation. The library is extendable without any complex forms and open to users to add, delete, or edit etc.

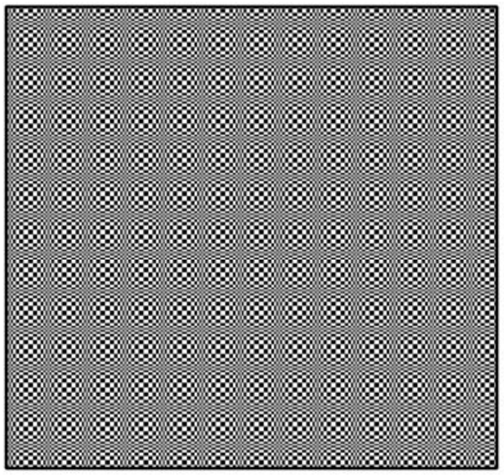
- **Product-transportable**
- The products of IPL can be stored in common file formats and can be used in most other applications. This feature ensures the language to be a general language for both pure image design and aid-oriented application design.

- **Robust**
- Image Processing Language is intended for writing programs that are reliable in a variety of ways. The language put emphasis on error checking and eliminating situations that are error prone.

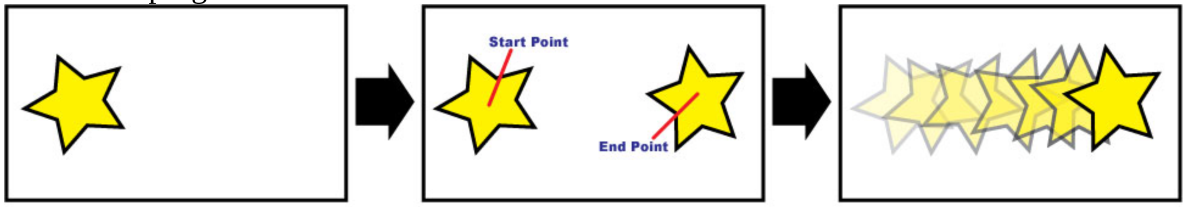
4. Intersting, representative program in our language

- Complex and recursive image making program
  ex) Esher pictures, repetitive pattern images, and tiles.

Circle of Limit III - Escher

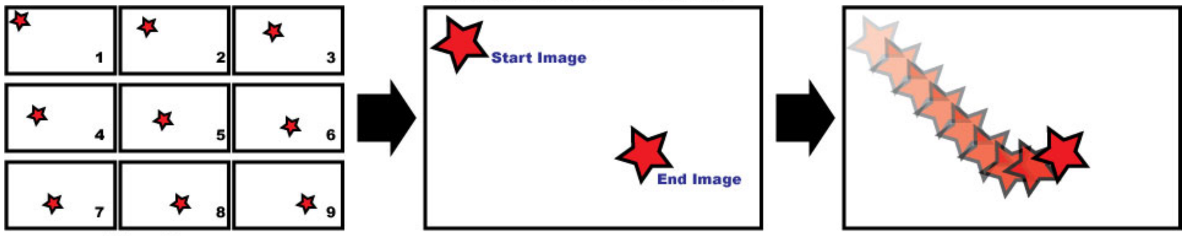

Pattern Image

- Simple animation making program
  ex) Flash-like simple animation making program with bitmap file, animation making program similar to animated GIF



Single Image

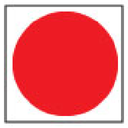
< Flash-like Animation >

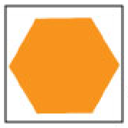


Bitmap Images

< GIF-like Animation >

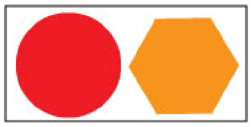
5. Examples of syntax



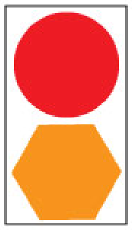


```
define imgA ./ex1.jpg;
define imgB ./ex2.jpg;
```

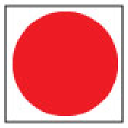
```
img1 = imgA + imgB;
```



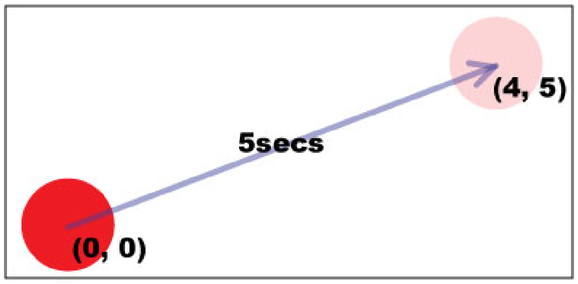
```
img2 = imgA & imgB;
```



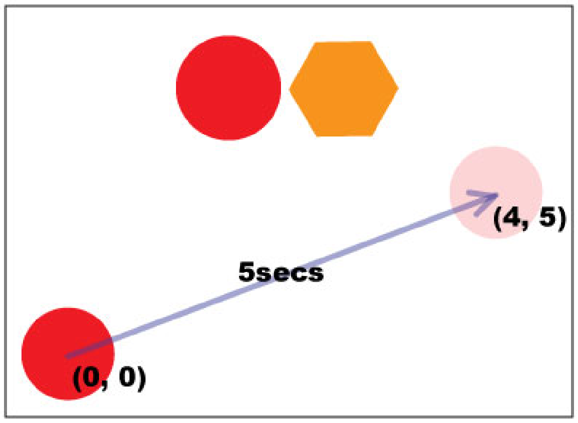
```
img3 = imgA;
```



```
img4 = move img2 (4, 5) 5 FADE_OUT;
```



```
display(img1&img4);
```



- defineImg <imgName> <imgPath> : define image object as <imgName> from <imgPath>.
  ex) defineImg imgA ./ex1.jpg
  - Define imgA as the image, from the 'ex1.jpg' file which is in the current folder.
- '+' operator : combine image horizontally. It can be used as "<imgObject> = <imgObject> + <imgObject> ". To illustrate, An example can be described a picture above.
- '&' operator : combine image vertically. It can be used as "<imgObject> = <imgObject> & <imgObject>". To illustrate, An example can be described a picture above.
- display (<imgObject>) : Display <imgObject> on to the screen. It is the only command that shows actual <imgName> to the screen after you set.
- set <imgObject> <coordination> <options>: set the <imgObject> to the <coordination> with <options>. <options> can be top-right, top-left, bottom-right, bottom-left, center. Default value for options is center.
  ex) set imgA (3,3) center
  - Set imgA to the (3,3) as a center.
- scale <imgObject> <scaleFactor> : magnify or reduce the <imgObject> with <scaleFactor>. <scaleFactor> is a floating value which represents a factor to magnify or reduce.
  ex) scale imgA 2
  - enlarge imgA as a factor of 2.
- move <imgObject> <destination> <options> : make <object> move from current position to <destination> following <options>. In <options>, there can be <duration>, <length>, <path>, etc.
  ex) move imgA (4,5)line
  - Move imgA following the line from current position to (4,5).
- animate <startImgObject> <endImgObject> <duration>
  ex) animate img00.jpg img10.jpg 5
  - Make an animation that starts with img00 and ends with img10, and time from start to end is 5 sec.
  - There should be a restriction on name of images for <startImg> and <endImg>. Only difference between them is number in their filename.
- rotate <imgObject> <degree> <options> : rotate <imgObject> <degree> to the <options>. In <options>, it could be clockwise or counterclockwise. The default for <options> is clockwise.
  ex) rotate imgB 30 counterclockwise
  - rotate imgB 30 degrees to the counterclockwise.