cs805 Assignment 2

Ray Shulang Lei
200253624
Department of Computer Science
University of Regina

October 22, 2012

Abstract

This assignment is written in literate programming style, generated by noweb, rendered by LaTex, and compiled by clang++ with c++ 11 standard.
assignment paper is at latex/as2.pdf
c++ programs are at src/*
binary executable for OS X 10.8 is inside bin

1 function implementation

```
<<src/util.cpp>>=
#include "util.h"
#include <math.h>

ImagePanel foreach_pixel_exec(ImagePanel img, std::function<int(Vector)> ray_func)
  int i = 0;
  for (auto& pixel: img) { //foreach pixel in empty_img
    pixel = ray_func({1.0,1.0,1.0});
    i++;
    std::cout<<i<<std::endl;
  }
  return img;
}

//initialize img panel to all 0s
ImagePanel init_img_panel(ImagePanel img) {
  for (auto& pixel: img) { //foreach pixel in empty_img
    pixel = 0;
  }
  return img;
}

//translate ray equation to an 0~255 shading value
int ray_tracing(Vector ray) {
  Intersection p = ray_objects_intersection(ray);
  return ray[0]+ray[1]+ray[2]+ray[3];
}

//calculate the ray object intersection point
Intersection ray_objects_intersection(Vector ray) {
  return {1,2,3,
          4,5,6,
          1.0};
}
```

```
//==========helpers==========

//Translate 2D array index of row column to 1D index.
//Notice that x, or column index, starts with 0.
//If return value is -1 then there is an out-of-bounce error.
int to_1d(int x, int y) {
  if (x >= IMG_X || x < 0)
    return -1;
  if (y >= IMG_Y || y < 0)
    return -1;
  return (IMG_Y*y + x);
}

//Translate 1d array index to 2d
std::array<int, 2> to_2d(int x) {
  if (x>=(IMG_X*IMG_Y) || x < 0) {
    return {-1,-1};
  }
  int y_ = x / IMG_X;
  int x_ = x % IMG_X;
  return {x_, y_};
}

//prints the img panel
void print_img_panel(ImagePanel img) {
  std::cout<<std::endl;
  for (auto& pixel : img) {
    std::cout<<pixel<<", ";
  }
  std::cout<<std::endl<<"Array size: "<<img.size()<<std::endl;
}

@
```

2 header

Here is an header file for typedefs and function declarations.

```
<<src/util.h>>=
#ifndef UTIL_H
#define UTIL_H

//define global vars
#define IMG_X 512
#define IMG_Y 512
#define IMG_LEN ( IMG_X * IMG_Y )

#include <array>
#include <functional>
#include <iostream>
typedef std::array<int, IMG_LEN> ImagePanel;
typedef std::array<float, 3> Point;
typedef std::array<float, 3> Vector;
typedef struct {
Point intersection; /* intersection point */
Vector normal; /* intersection polygon normal vector */
float kd; /* diffuse reflection coefficient of the surface */
} Intersection;

ImagePanel foreach_pixel_exec(ImagePanel, std::function<int(Vector)>);
ImagePanel init_img_panel(ImagePanel);
int ray_tracing(Vector);
Intersection ray_objects_intersection(Vector);

//helpers
int to_1d(int, int);
std::array<int, 2> to_2d(int);
void print_img_panel(ImagePanel);
#endif
@
```

3 main funciton

```
<<src/main.cpp>>=
#include <iostream>
#include <typeinfo>//debugging only
#include "util.h"

int main () {
  ImagePanel resultImg;
  resultImg = init_img_panel(resultImg);
  resultImg = foreach_pixel_exec(resultImg, [](Vector x){return ray_tracing(x);});
  print_img_panel(resultImg);

  //unit tests
  std::cout<<to_1d(0, 1)<<std::endl;
  std::cout<<to_2d(512)[0]<<std::endl;
  std::cout<<to_2d(512)[1]<<std::endl;
  std::cout<<to_1d(1, 1)<<std::endl;
  std::cout<<to_2d(513)[0]<<std::endl;
  std::cout<<to_2d(513)[1]<<std::endl;
  std::cout<<to_1d(511, 1)<<std::endl;
  std::cout<<to_2d(1023)[0]<<std::endl;
  std::cout<<to_2d(1023)[1]<<std::endl;
  std::cout<<to_1d(512, 1)<<std::endl;
  std::cout<<to_2d(512*512)[0]<<std::endl;
  std::cout<<to_2d(512*512)[1]<<std::endl;
  return 0;
}
@
```

4 compile script

Furthermore, this is the command to link these files. Notice that I am using -std=c++11 flag to enable c++ 11 features. The output binary executable is bin/run

```
<<compile.sh>>=
clang++ -std=c++11 -stdlib=libc++ -o bin/run src/main.cpp src/util.cpp
@
```