UNIVERSIDAD DE BUENOS AIRES
FIUBA

66.20 Organización de Computadoras
Trabajo práctico 0: Infraestructura básica
1^{er} cuatrimestre de 2018

ALUMNOS

92454 ZARAGOZA, MARTIN
92691 SIBIKOWSKI, NICOLAS
91985 DUFAU, EZEQUIEL

1. Introdución

Este documento representa la documentación técnica del trabajo práctico 0, correspondiente a la materia **66.20 Organización de Computadoras** En el mismo se incluira el desarrollo de los siguientes puntos:

- Diseño e Implementación
- Compilación y ejecución del programa
- Corridas de prueba
- Código Fuente

2. Diseño e Implementación

Se desarrolló un programa que permite dibujar el conjunto de Julia, en lenguaje C.

2.1. Tipos de Datos Abstractos

Se diseñaron los siguientes TDAs:

2.1.1. Pixel

Define un punto en el plano complejo

```
typedef struct {
unsigned x;
unsigned y;
} Pixel;

typedef struct {
unsigned width;
unsigned height;
} Dimension;

typedef Dimension Resolution;

void parseRes(char* str , Resolution* targetRes);

#endif
```

2.1.2. Complex

Define el TDA para manejar números complejos

```
typedef struct {
double re;
double im;
} Complex;

typedef struct {
double left;
double right;
double top;
double bottom;
} Boundaries;

Complex newCpx(double re , double im);

/**
* Obtiene un numero complejo a partir de un string
*/
void parseCpx(char* str , Complex* targetCpx);

/**
* Obtiene la raiz cuadrada de un complejo
*/
Complex pow2Cpx(Complex* value);

Complex addCpx(Complex* first , Complex* second);

/**
* Obtiene el modulo de un valor complejo
  (distancia respecto al origen).
*/
double modCpx(Complex* value);

/**
* Calcula los limites del plano complejo
   (el rectangulo a mapear del plano complejo).
* Dimension dim - Tamano del plano complejo
* Complex center - Valor de centro EN EL plano
*/
```

```
Boundaries getBoundaries(Complex* dim , Complex* center);

#endif
```

3. Compilación y ejecución del programa

Utilizamos el programa GXemul para simular un entorno de desarrollo. El mismo consta de una máquina MIPS corriendo una versión del sistema operativo BSD. Debido a esto, el compilador utilizado utilizado es el que trae instalada la imagen provista por la cátedra. El mismo es el **(GCC) 3.3.3 (NetBSD nb3 20040520)**

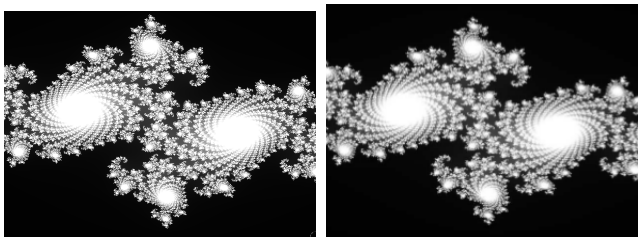
Para facilitar el proceso de compilación, programamos un Makefile, seteando las opciones correspondientes. Se debe ejecutar make en el directorio raiz del proyecto y el programa queda compilado. Se genera un ejecutable llamado app.exe.

4. Corridas de prueba

4.1. Valores por defecto

Se generó un dibujo usando los valores por defecto, barriendo la región rectangular del plano comprendida entre los vértices -2+2i y+2-2i. La validación fue visual. Comparando contra un dibujo patrón disponible en el enunciado.

Comando ejecutado: tp0 -o uno.pgm Resultado de la prueba:



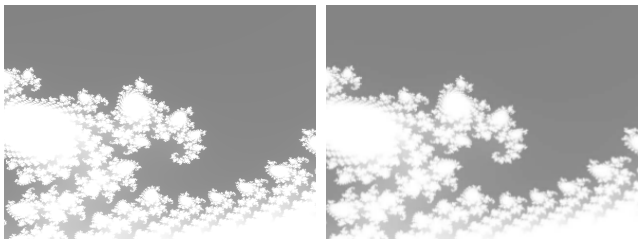
(a) nuestro programa (b) trabajo práctico

Figura 1: Comparación entre imagen generada por el enunciado del tp, y nuestro programa

4.2. Zoom en región

Hacemos zoom sobre la región centrada en 0.282-0.007i, usando un rectángulo de 0.005 unidades de lado.

Comando ejecutado: tp0 -c 0.282-0.007i -w 0.005 -H 0.005 -o dos.pgm
Resultado de la prueba:



(a) nuestro programa (b) enunciado del trabajo práctico

Figura 2: Comparación entre la imagen generada por el enunciado del tp, y nuestro programa

Indice