Universidad de Buenos Aires

Taller de Programación (75.42)

Filtro Morfológicos

Cristian González

94719

Índice

1. Clases **2**
- 1.1. Interpreter . . . 2
- 1.2. Filter . . . 2
- 1.3. Erosion . . . 2
- 1.4. Dilatation . . . 2
- 1.5. Matrix . . . 2
- 1.6. Position . . . 2
- 1.7. Thread . . . 2
- 1.8. FiltrosMorfologicos . . . 2

2. Diagramas **3**

3. Dificultades **4**

4. Cambios **4**

5. Links **5**

1. Clases

1.1. Interpreter

La responsabilidad de esta clases es dado en un string que dentro tiene el formato de una matriz, poder interpretarlo y crear una matrix con el formato indicado.

1.2. Filter

Es una clase abstracta que se encarga de dado una matrix imagen y una matrix patrón encargarse de constatar una matrix con la otra, tiene el compartimiento general de los filtros del trabajo práctico.

1.3. Erosion

Modela el comportamiento del Filtro Erosion que busca la coincidencias totales para emitir en la matrix destino.

1.4. Dilatation

Se encarga de modelar el comportamiento del Filtro Dilatación y de tal manera buscar coincidencia parciales para luego emitir en la matrix destino.

1.5. Matrix

Es el repositorio de la ïmagenes"tanto el de origen, patrón y destino, básicamente es una matrix que contiene strings.

1.6. Position

Simplemente modela las coordenada de una matrix (fil,col) y realiza operaciones básica como suma de posiciones como también obtener la posición relativa.

1.7. Thread

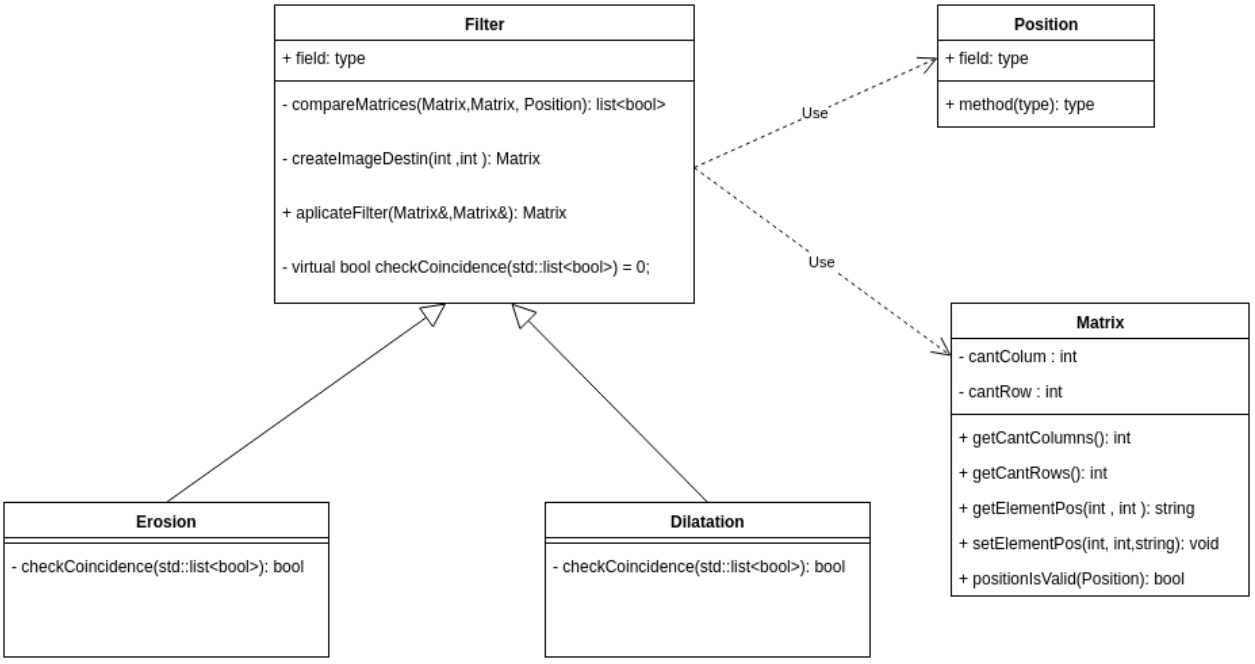
Se encarga de de modelar el comporamiento de un thread, para crearlo se tiene que proveer los datos que va a necesitar ese thread como la función/método a ejecutar.

1.8. FiltrosMorfologicos

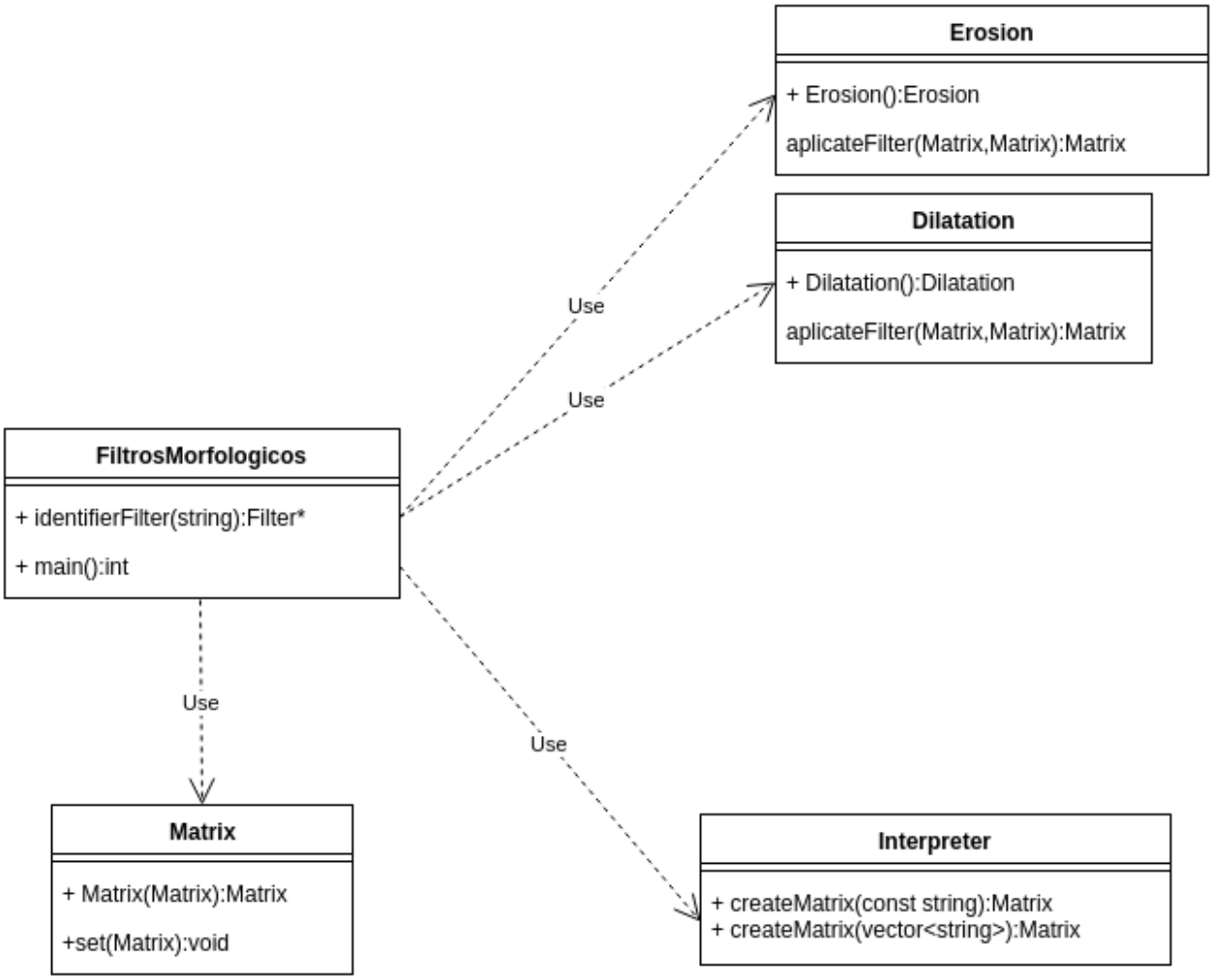
Clase punto de entraga al sistema, se encarga de procesar lo que se ingresa por STDIN, como controlar la cantidad de hilos y aplicar de forma secuencial los filtros que correspondan.

2. Diagramas

Acá se muestra como se relaciona el filtro con las otras clases, el Filter utiliza a la Matrix para poder conocer sus dimensiones, poder crear Matrix vacía, como poder obtener elemento de cierta Position.
Tanto como Dilatation como Erosion definen el método `checkCoincidence()` ya que cada filtro busca distintas coincidencias, esto es conocido como el patrón Template Method.



La clase principal interactúa con 3 clases:Filter,Interpreter y Matrix. Con Interpreter se encarga de identificar qué clase de Filter a aplicar y crea un instancia de la misma, para luego aplicar un filtro a la Matrix ingresada por STDIN junto con el patrón devuelto por Interpreter. A Matrix lo utiliza para poder crear una instancia que se copiar a partir de otra y poder setear todos los valores de una Matrix a partir de una instancia que se le pasa.



3. Dificultades

1. Al crear un Matrix utiliza `malloc()` para poder reservar memoria, eso generaba `jump conditional` que si bien es memoria reservada, puede tener basura.
2. Al utilizar el patrón Template Method, la idea inicial era poder simplemente conocer un Filter y aplicar el filtro, pero para eso tuve que recurrir al `new`, ya que sí o sí necesito un puntero para que llame al método definido en la clase base.

4. Cambios

- En Matrix hay un método set, este en realidad debería ser una sobrecarga del `operator =` siendo un `asignación por movimiento`.
- El método `createMatrix()` debería ser un `constructor por movimiento`.

- El método `aplicateFilter()` debería ser un `constructor por movimiento`.
- Para los `Threads` la idea es crear un container que tenga los datos necesarios para `Threads`, y que sea accesible por `Singleton`.

5. Links

Link del repositorio:https://github.com/Cristian3629/Filtros-Morfologicos