IMPLEMENTATIEPLAN SCALING (NEAREST-NEIGHBOR)

NAMEN EN DATUM

Robert Bezem en Jos Bijlenga

02-06-2015

DOEL

Het doel van deze implementatie is een werkend Nearest-Neighbor algoritme te implementeren zodat de snelheid hiervan getest kan worden.

METHODEN

Voor Nearest-Neighbor is er eigenlijk maar één methode. De pixels waarvan de waarde na het schalen niet bekend is worden opgevuld met de waarde van een van de omringende pixels.

KEUZE

Om de nieuwe grote van het plaatje te bewaren is er voor gekozen om de verhouding van de originele afbeelding te behouwen. Dit zorgt ervoor dat gezichtskenmerken niet uitgerekt of ingedrukt worden.

```
const int newX = image.getWidth()*0.5, newY = newX / ((float)image.getWidth() / (float)image.getHeight());
```

IMPLEMENTATIE

```
	const int newX = image.getWidth()*0.5, newY = newX / ((float)image.getWidth() / (float)image.getHeight());
	IntensityImage * newImage = ImageFactory::newIntensityImage(newX, newY);
	float x_ratio = (float)image.getWidth() / newX;
	float y_ratio = (float)image.getHeight() / newY;
	int x2, y2;
	for (int x = 0; x<newX; x++) {
		for (int y = 0; y<newY; y++) {
			x2 = floor(x*x_ratio);
			y2 = floor(y*y_ratio);
			newImage->setPixel(x, y, image.getPixel(x2, y2));
		}
	}
```

Hier is duidelijk te zien dat er eerst wordt berekend hoe groot de nieuwe afbeelding moet zijn. Dan wordt berekend wat de dichtstbijzijnde pixel is d.m.v. een pixelratio en dan wordt de nieuwe pixel de kleur van een van de oude pixels gegeven.

EVALUATIE

Om de implementatie te testen zullen er verschillende afbeeldingen worden getest met verschillende schalingsfactoren. Hier wordt daarna gekeken of er een juiste afbeelding uit komt.