1. Implementatieplan titel Week5 Scaling

1.1. Namen en datum

Martijn van der Struijk & Tim Hasselaar

1-06-2015

1.2. Doel

Het doel van deze opdracht is om een afbeelding naar een goed formaat te scalen. Dit kan soms noodzakelijk zijn als een bedrijf of iets dergelijks alleen maar met een bepaald formaat werkt.

1.3. Methoden

1. nearest neighbor (zero-order)

2. bilinear interpolation (first-order)

3. bicubic (high-order)

1.4. Keuze

Wij hebben bij deze opdracht gekozen voor de bilinear interpolation methode. Deze methode ligt tussen beide methodes in en dat leek ons wel voldoende voor deze opdracht. Daarnaast is de bicubic computationeel duur maar wel veel mooier en de nearest neighbor blokkerig maar wel snel.

1.5. Implementatie

We bepalen aan de hand van de scaling ratio een positie binnen de pixels van het plaatje dat gescaled moet worden. Van die positie pakken we de 4 dichtstbijzijnde pixels daar pakken we een gewogen gemiddelde van. Dit gemiddelde wordt de nieuwe pixel waarde voor het gescalde plaatje.

```
for (int i = 0; i < new_height; i++){
         for (int j = 0; j < new_width; j++){
            int x, y;
            double x_difference, y_difference;
            int index;
            x = (int)(j / rescale_ratio);
            y = (int)(i / rescale_ratio);
            x_difference = (j / rescale_ratio) - x;
            y_difference = (i / rescale_ratio) - y;
            index = y * (int)old_width + x;

            int A = image.getPixel(index) & 0xff;
            int B = image.getPixel(index + 1) & 0xff;
            int C = image.getPixel(index + (int)old_width) & 0xff;
            int D = image.getPixel(index + (int)old_width + 1) & 0xff;

            // De formule is: A ( 1 - w ) ( 1 - h ) + B * w * ( 1 - h ) + C * h * ( 1 - w
) + D * w * h;
            int final_value = (int)(
               A * (1 - x_difference) * (1 - y_difference) +
```

```
                B * x_difference * (1 - y_difference) +
                C * y_difference * (1 - x_difference) +
                D * x_difference * y_difference
                );
            i_image->setPixel(j, i, final_value);
        }
    }
}
```

1.6. Evaluatie

We gaan aan de hand van een ingebouwde timer kijken of onze code sneller of langzamer is dan de originele code. We zullen ook kijken naar het gescalde plaatje of die aan de nieuwe eisen voldoet. We zullen ook onze manier van scalen vergelijken met die van de originele code.