
Team 12

Frédéric Blondeel	26h
Martijn Debeuf	35h
Toon Sauvillers	h
Dirk Vanbeveren	21h
Bert Van den Bosch	h
Seppe Van Steenberghe	1h

Inhoudsopgave

1	Introductie	2
2	Remap	2
3	Foto naar client sturen	4
4	Testen	4
5	Valkuilen	4
6	Besluit	4

1 Introductie

Nu de schermen gevonden en geïdentificeerd zijn kan men wat meer dan lijntjes displayen. Met deze taak zullen verschillende schermen gebruikt worden om een grotere foto te tonen. Het is dus mogelijk om enkele gsm's samen één image te tonen, verspreid over de displays. Hiervoor wordt gebruik gemaakt van perspectief veranderende matrices, server/client communicatie, etc.

2 Remap

Doordat schermen in 3d gedraaid zijn is er nood aan een correctie van het perspectief. Dit is onder meer nodig voor het lezen van de barcode en het displayen van foto's. Het basis algoritme die hiervoor gebruikt wordt zal source en destination corners gebruiken als input. Deze corners zijn de start en eind hoeken, de foto zal dus naar de destination corners worden gerokken. Er zal per corner array een matrix worden berekend. Eerst worden de coëfficiënten berekend.

$$\begin{bmatrix} x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} \lambda \\ \mu \\ \tau \end{bmatrix} = \begin{bmatrix} x_4 \\ y_4 \\ 1 \end{bmatrix}$$

$$\begin{bmatrix} \lambda \\ \mu \\ \tau \end{bmatrix} = \begin{bmatrix} x_1 & x_2 & x_3 \\ y_1 & y_2 & y_3 \\ 1 & 1 & 1 \end{bmatrix}^{-1} \begin{bmatrix} x_4 \\ y_4 \\ 1 \end{bmatrix}$$

Daarna wordt de 3x3 matrix geschaald met de gevonden coëfficiënten.

$$\begin{bmatrix} \lambda x_1 & \mu x_2 & \tau x_3 \\ \lambda y_1 & \mu y_2 & \tau y_3 \\ \lambda & \mu & \tau \end{bmatrix}$$

De twee matrices A en B, respectievelijk met de source en destination corners, worden gebruikt om de transformatie matrix te berekenen.

$$C = AB^{-1}$$

Deze zal elke destination pixel een de source pixel geven aan de hand van:

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = C \begin{bmatrix} x \\ y \\ 1 \end{bmatrix}$$

$$x_{nieuw} = x'/z'$$

$$y_{nieuw} = y'/z'$$

Een simpel algoritme zal over alle pixels (dit zijn width * height pixels) gaan en de correcte kleur toevoegen. Zie figuur 1.[1]

Remap Test



Figuur 1: Voorbeeld van een remapped foto

3 Foto naar client sturen

In de Screen class, is er een functie dat van een canvas in de grootte van de originele foto het gedeelte van zijn eigen scherm kan knippen en transformeren met perspective transformation. Er is dus een canvas met de juiste grootte op het scherm waarop een foto op geüpload kan worden. Deze foto kan verschoven worden met de muis. Eens dat de foto op de gewilde plaats staat wordt elke scherm er uit geknipt, getransformeerd en verzonden naar de juiste client.

4 Testen

De calculaties van de remap functie zijn simpelweg getest met een kleine foto (figuur 1).

5 Valkuilen

6 Besluit

Referenties

- [1] Martin von Gager. Redraw image from 3d perspective to 2d. <https://stackoverflow.com/questions/14244032/redraw-image-from-3d-perspective-to-2d>, 2016.