— Verslag P&O CW 2019–2020 Taak 7 —

Department Computerwetenschappen – KU Leuven 12 november 2019

Team 12	
Frédéric Blondeel	26h
Martijn Debeuf	35h
Toon Sauvillers	h
Dirk Vanbeveren	21h
Bert Van den Bosch	h
Seppe Van Steenbergen	1h

Inhoudsopgave

1 Introductie

Nu de schermen gevonden en geidentificeerd 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 world 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 algorithme 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 berekent. Eerst worden de coefficienten berekent.

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

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

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

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

De twee matrices A en B, respecitivelijk 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:

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

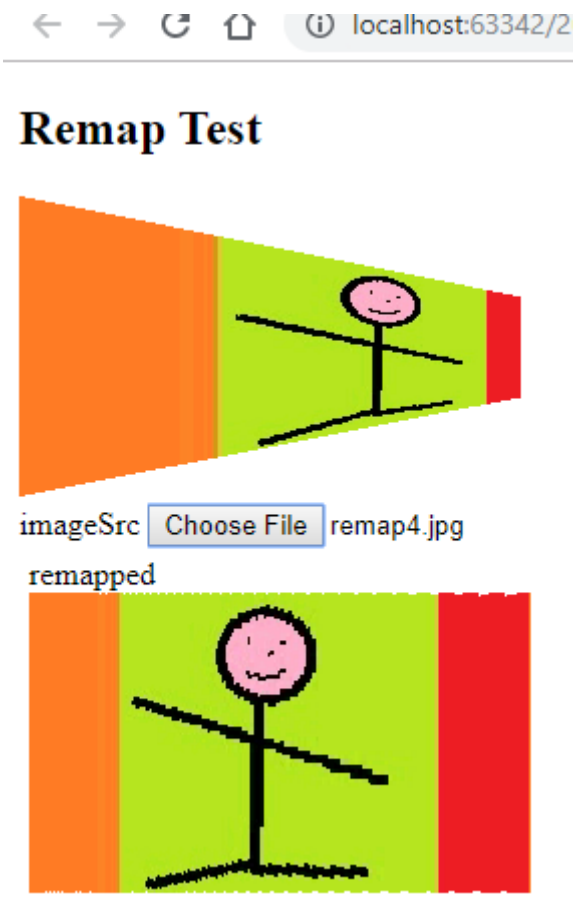
$$x_{nieuw} = x'/z'$$

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

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

3 Testen

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



Figuur 1: Voorbeeld van een remapped foto

4 Valkuilen

5 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.