|  |  |
| --- | --- |
|  | |
|  |  |
| aan | Alessandro Ledda |
| van | Geert d’Hoine, Hans Van den Broeck |
| datum | 17-05-2010 |
| pagina’s |  |
|  | |
| **Verslag Multimedia 4: Beeldverwerking 4** | |
|  | |

[Probleemstelling 3](#_Toc262181754)

[Onderzoeksvraag 3](#_Toc262181755)

[Voorgestelde Oplossingen 3](#_Toc262181756)

[Het roteren 3](#_Toc262181757)

[Vector genereren van 95 bits 3](#_Toc262181758)

[Het decoderen van de bitvector 3](#_Toc262181759)

[Bespreking Grafische Interface 5](#_Toc262181760)

[Besluit 6](#_Toc262181761)

[Literatuurlijst 7](#_Toc262181762)

# Probleemstelling

## Onderzoeksvraag

Hoe leest men een streepjescode in uit een afbeeldingsbestand, dat eventueel geroteerd is, met Matlab, zonder gebruik te maken van optische karakterherkenning, dus alleen gebruik makend van de streepjes zelf?

# Voorgestelde Oplossingen

Om het werk wat eenvoudiger te maken hebben we het probleem in deelproblemen opgedeeld:

* Het roteren van de afbeelding zodat de streepjes van de streepjescode verticaal zijn geörienteerd.
* De streepjescode destilleren uit de afbeelding, zodat we een vector verkrijgen van 95 bits.
* Berekenen hoe de barcode gedecodeerd is (welke pariteit wordt er gebruikt).
* De barcode decoderen aan de hand van de berekende pariteit.

## Het roteren

De afbeelding wordt eerst omgezet naar een 1-bit (zwart-wit) bitmap. Daarna worden de randen gedetecteerd met het Canny randdetectiealgoritme.

Op deze bitmap wordt dan een radon-transformatie uitgevoerd. Uit het resultaat zoeken we dan de hoogste waarde. Het x-coördinaat van deze komt overeen met de hoek die nodig is om de streepjescode recht te draaien.

Ten slotte wordt de afbeelding geroteerd over deze hoek.

## Vector genereren van 95 bits

De afbeelding uit de vorige stap wordt nu herleid naar een ééndimensionale vector. Deze vector wordt geïnverteerd, (alle ‘1’ worden ‘0’, alle ’0’ worden ‘1’). In het resultaat wordt dan gezocht naar de eerste en de laatste zwarte pixel. De coordinaten van deze pixels worden bijgehouden. Daartussen ligt onze streepjescode.

Onze vector wordt nu bijgesneden, zodat deze enkel de streepjescode bestaat. Deze vector heeft nu wel een lengte die een geheel veelvoud is van 95. Nu kunnen we deze vector herschalen, zodat de lengte exact 95 wordt.

## Het decoderen van de bitvector

De decoder gaat ervan uit dat de streepjescode voldoet aan de EAN-13 standaard.

De guard-bits op het begin, het midden en het einde van de streepjescode worden genegeerd.

Eerst proberen we te bepalen wat het eerste cijfer van de code is, want dit cijfer is niet inbegrepen in de streepjescode, en is wel noodzakelijk om de barcode achteraf te decoderen.

De eerste zes cijfers van de streepjescode zijn gecodeerd aan de hand van een pariteit die bepaald wordt door het eerste cijfer (het cijfer voor de eigenlijke streepjescode). We tellen het aantal eentjes die worden gebruikt per cijfer in de streepjescode. Een even pariteit wordt voorgesteld door de letter ‘G’, een oneven pariteit door de letter ‘L’.

Onderstaande Matlab-code geeft het verband aan tussen de pariteit van de cijfers en het cijfer dat voor de streepjescode staat:

Figuur 1: Extract uit decodeEan.m (Lijnen 120-22)

colorMap = containers.Map({'LLLLLL', 'LLGLGG', 'LLGGLG', ...

'LLGGGL', 'LGLLGG', 'LGGLLG', 'LGGGLL', ...

'LGLGLG', 'LGLGGL', 'LGGLGL'}, ...

{0, 1, 2, 3, 4, 5, 6, 7, 8, 9});

Nu weten we hoe de streepjescode gecodeerd is. Onderstaande code geeft dit verband weer (de laatste zes cijfers hebben altijd allemaal de codering ‘R’):

Figuur 2: Extract uit decodeEan.m (Lijnen 40-56)

% blue color map

colorLmap = containers.Map({'0001101', '0011001', '0010011', ...

'0111101', '0100011', '0110001', '0101111', ...

'0111011', '0110111', '0001011'}, ...

{0, 1, 2, 3, 4, 5, 6, 7, 8, 9});

%green color map

colorGmap = containers.Map({'0100111', '0110011', '0011011', ...

'0100001', '0011101', '0111001', '0000101', ...

'0010001', '0001001', '0010111'}, ...

{0, 1, 2, 3, 4, 5, 6, 7, 8, 9});

% red color map

colorRmap = containers.Map({'1110010', '1100110', '1101100', ...

'1000010', '1011100', '1001110', '1010000', ...

'1000100', '1001000', '1110100'}, ...

{0, 1, 2, 3, 4, 5, 6, 7, 8, 9});

De functie getColor(position) geeft de kleur van het bitpatroon dat aan de beurt is. Elk patroon van zeven opeenvolgende bits wordt opgezocht in de juiste associatieve lijst ( zie hierboven). Dit geeft ons de overeenkomstige decimale waarde. Dit is dus het cijfertje dat bij het zeven-bit-patroon hoort. Dit cijfer is ook onder de streepjescode terug te vinden.

Als laatste stap wordt de controlesom berekend om na te gaan of er niets mis is gegaan. Tijdens het testen werd de controlesom altijd correct berekend.

## 

## Bespreking Grafische Interface

Eerst duwt de gebruiker op de knop ‘Load Image’. Er moet naar een afbeelding gebladerd worden. Deze afbeelding wordt dan geladen.

Daarna klikt de gebruiker op de knop ‘Read Barcode’. De streepjescode wordt geroteerd, omgezet naar een binaire vector en gedecodeerd. De tussenstappen worden getoond.

De gelezen code (decimaal) verschijnt rechts.



# Besluit

Hetgeen we geprobeerd hebben lukt perfect. Al wordt er bij het decoderen een beetje dubbel werk gedaan. Deze code werd eerst geschreven met de aanname dat de gebruiker zelf het eerste cijfer manueel invoert, zodat de coderingswijze bekend is en de streepjescode kan gedecodeerd worden.

Later hebben we dan een methode bedacht om dat eerste cijfer te achterhalen. Het resultaat van deze functie vervangt dan de invoer van de gebruiker.

# Literatuurlijst

* Canny edge detector:
  + Green, B (2002). Canny Edge Detection Tutorial. Drexel University, PH, USA.   
    Beschikbaar op <http://www.pages.drexel.edu/~weg22/can_tut.html> [21-05-2010]
  + Tai, S.L. (2002). Canny Edge Detector. Drexel University, PH, USA.   
    Beschikbaar op <http://www.pages.drexel.edu/~pyo22/students/designTeams/kite2001WorkFolder/cannyEdgeDetector.pdf> [21-05-2010]
  + Moeslund T. (2009). Canny Edge Detection. Universiteit van Aalborg, DK.   
    Beschikbaar op <http://www.cvmt.dk/education/teaching/f09/VGIS8/AIP/canny_09gr820.pdf> [21-05-2010]
* Radon transformatie:
  + Onbekend (2010). Radon Transform. Wikipedia.   
    Beschikbaar op <http://en.wikipedia.org/wiki/Radon_transform> [21-05-2010]
  + Toft, P (1996). The Radon Transform. Techincal University of Denmark, DK.   
    Beschikbaar op <http://eivind.imm.dtu.dk/staff/ptoft/Radon/Radon.html> [21-05-2010]
  + Onbekend (2010). Radon Transform. The MathWorks, inc. Beschikbaar op <http://www.mathworks.com/access/helpdesk_r13/help/toolbox/images/transfo9.html> [21-05-2010]
  + Xiao, S., Wu, Y. (2006). Rotation-invariant Texture Analysis Using Radon and Fourier Transforms. IOP Publishing.  
    Beschikbaar op <http://iopscience.iop.org/1742-6596/48/1/268/pdf/jpconf7_48_268.pdf> [21-05-2010]
* Varia:
  + Associatieve lijst in Matlab  
    Onbekend (2010). Map Containers. The MathWorks, inc.   
    Beschikbaar op: [http://www.mathworks.com/access/:helpdesk/help/techdoc/matlab\_prog/brqqo5e-1.html](http://www.mathworks.com/access/helpdesk/help/techdoc/matlab_prog/brqqo5e-1.html) [21-05-2010]
* Bronnen voor code:
  + Lynne.C, Tse. C. M., Lokuvithana. Dilak, Lee. V. (2006).   
    Barcode image recognition. Softwarepractice.org  
    Beschikbaar op: http://www.softwarepractice.org/wiki/Barcode\_Image\_Recognition