



---

# Individueel Rapport

---

27 juni 2018

*Student:*  
Abdelilah Ahbari  
12021954

*Tutor:*  
Toto van Ine  
*Practicumgroep:*  
C2

*Cursus:*  
multimedia

## 1 Introductie

Voor het groepsproject van Multimedia is er gekozen om een applicatie te maken die een Sudoku in een foto kan herkennen door middel van OCR (Optical Character Recognition).

Na het herkennen wordt de sudoku in een array van 81 elementen gestopt (Hier later meer uitleg over).

De gebruiker krijgt dan de sudoku op het scherm te zien en kan dan een aantal dingen doen: De sudoku zelf oplossen, dit kan door middel van ouderwets met de toetsenbord of spraak gestuurd, of hij kan ervoor kiezen de sudoku te laten oplossen door de geïmplementeerde algoritme.

## 2 Methode

Als eerst heb ik geprobeert de sudoku solver algoritme te implementeren door middel van een brute force algoritme.

Deze algoritme probeert alle mogelijke combinaties op elke lege cell. Zodra hij dan een cell tegen komt waar hij alle getallen (1 t/m 9) heeft geprobeert en deze allemaal niet kunnen vanwege de regels van een sudoku zal de algoritme 1 cell terug gaan en daar het volgende getal proberen. Dit gaat zo door totdat het algoritme een oplossing heeft gevonden, of uitvindt dat de sudoku onoplosbaar is.

Na het implementeren van deze algoritme zijn wij tot de conclusie gekomen dat het handiger was om een snellere en een efficiëntere algoritme te gebruiken. Omdat het doel onorigineel was om de opgeloste sudoku op de camera feed te laten zien.

Ik heb na deze beslissing een slimmere algoritme geïmplementeerd. zie Algoritme voor meer informatie.

### 2.1 Algoritme

Het nieuwe slimmere algoritme is gebaseerd op het brute force algoritme. Door het toevoegen van een aantal regels is deze gemiddeld 5x sneller gemaakt dan het origineel.

Het algoritme heeft als eerste structureel verschil dat het een single Array gebruikt (t.o.v. de double Array van de brute force). Hier heb ik voor gekozen omdat het een stuk sneller

is om data uit een single array te halen. Het tweede structureel verschil is dat alle functies van het 2de algoritme static zijn, hiervoor heb ik gekozen omdat het sneller en efficiënter is om static functies aan te roepen ten opzichte van non-static functions.

Naast de structurele verschillen zitten er ook verschillen in het algoritme.

- Het nieuwe algoritme gebruikt 2 lijsten 1 om de ingevulde waarden in de cellen op te slaan en 1 om alle mogelijke waarden op te slaan aan de hand van de al ingevulde waarden.
- Het nieuwe algoritme vult de sudoku ook niet gewoon van links naar rechts in. Het nieuwe algoritme vindt eerst alle cellen waar maar 1 waarde mogelijk is en die vult hij in.
- De sudoku die je dan hebt wordt als "True" beschouwd (Het originele meegegeven sudoku wordt hiermee vervangen).
- Dan worden de rest van de cellen geordend van minst mogelijke tot meest mogelijke opties.
- Dan worden deze cellen ingevuld aan de hand van het principe van het eerste algoritme
- Elke keer wanneer een nieuwe waarde wordt ingevuld wordt het 2de lijst, die alle mogelijke gegevens heeft, aangepast aan de hand van de zojuist ingevulde waarde.

## 2.2 Git

Ook was ik verantwoordelijk voor het merge van de verschillende branches. Wij hebben ervoor gekozen om met merge requests te werken omdat er dan minder snel fouten en errors in de master branch belanden. Daarnaast hebben we gekozen om mij deze taak te geven omdat ik het meeste ervaring heb met git en al eerder had gewerkt met merge requests. Omdat het project maar 2 weken in beslag nam, hebben wij gekozen om niemand anders deze taak te geven omdat hij dan te weinig tijd had om dit te begrijpen.

## 3 Resultaten

Door het implementeren van het 2de algoritme is het oplossen van een sudoku gemiddeld 5x zo snel geworden. Hierdoor is het goed mogelijk om de opgelost sudoku live weer te geven op de camera feed. Alhoewel dat onze originele plan was zijn wij erachter gekomen dat dit te veel was om in 2 weken af te krijen.