In dit document staat beschreven wie wat voor sprint 4 heeft gedaan binnen onze groep.

**Bryan**: Ik heb samen met Chanan van Ooijen de klassen Equalize, Histogram, LineDetection, MedianFilter en Threshold gemaakt. Dit hebben wij in de tijd van vrijdag 13:00 – 18:00, 20:00 – 3:00 en zaterdag van 13:00 – 10:00 (Ja, de nacht is doorgewerkt). Yusuf Syakur heeft wel eerst de Threshold klasse gemaakt door middel van het implementeren van de OTSU methode, maar dit hebben Chanan en Ik samen herschreven omdat de OTSU methode niet de gewenste waarden gaf. Ook heb ik de hoofdstukken: “De te testen algoritmen – Algoritme 2” en “Hoe is de test verlopen + resultaten” geschreven.

**Lars**:

**CannyEdge algorithme** puur gemaakt voor testredenen. (uiteindelijk niet gebruikt).  
**Binary Yellow algorithme** Controleert op een ingestelde ‘geel’ range.   
**XMLReader** Leest filenames en attributen in het XML bestand.  
 Verder zit hier een controleer functie in.   
 (Gevonden hoekpunten V gegeven waardes)  
**Automatische tester**: Heb ook een stuk code geschreven zodat uit de xml files de te controleren fotos gepakt worden. Hiermee word vervolgens doorheen gelooped.   
**Modificatie** op DetermineCornerPoints( in LineDetection) zodat het een array terug geeft met de gevonden punten.  
  
Verder hoop onderzoek gedaan naar hough transform en trigonometry. Dit is vervolgens niet in het project opgenomen. Verder heb ik niks gedaan aan het verslag.

**Yusuf**:

Sobel Edge algoritme   
Deze heb ik gemaakt omdat het een onderdeel was van onze eerste algoritme, het zou na de MedianFilter moeten worden gedaan en hierop volgt de Threshold methode. Vanuit de intensiteitshistogram van de resultaat zou de intensiteit hoog moeten zijn bij de kenteken waardoor lokalisatie mogelijk was. Helaas was dit niet het geval en was mijn werk voor niks geweest.

OTSU Threshold algoritme  
Ik pastte OTSU aan, omdat ik niet wist hoe ik een goede threshold zou moeten krijgen dat werkt voor alle plaatjes. Op internet heb ik gekeken naar de beste opties en OTSU kwam eruit. Het was wel goed geimplementeerd maar de volgorde van onze werkwijze voor de 1e algoritme zat fout. Bryan besloot deze weg te halen.

Verslag  
Ik heb de te testen algoritmen – algoritme 1 herschreven met afbeeldingen. Zo werd een duidelijke beeld gegeven van wat moest gebeuren maar niet lukte.

**Chanan**:  
Ik heb samen met Bryan Baan de klassen Equalize, Histogram, LineDetection, MedianFilter en Threshold (versie van Yusuf herschreven) gemaakt. Vrijdag 07/03/2014 heb ik daar aan gewerkt van 11:00 – 17:00 en zaterdag van 10:00 – 16:00 en 23:00 tot zondagochtend 06:00. “De te testen algoritmen – Algoritme 1” en “Wat gaan we testen” heb ik geschreven.

**Mitchell**:

Ik heb samen met Yusuf de sobelfilter en de treshold filter geschreven. Daarnaast hebben we code geprobeerd om automatisch een ‘juiste’ treshold te bepalen. Ik heb Bryan geholpen met de median filter maar uiteindelijk heeft hij deze zelf herschreven met Chanan. Ik heb zelf geprobeerd een color filter te schrijven die probeert het kenteken te vinden door de RGB kleuren om te zetten naar het HSV formaat. Vervolgens kun je dan op hue filteren maar het algoritme pakte teveel kleuren waardoor hij niet alleen het kenteken vond. Het is me uiteindelijk niet op tijd gelukt om te vinden waar de fout zat daarom is de code niet terug te zien in onze opgeleverde code.