Low-level feature detectie (practicum 2)

We hebben nu gebruik gemaakt van onze kennis over objecten om deze snel te kunnen detecteren en aan een algoritme (in dit geval een auto) door kunnen geven. Wat zou er nu gebeuren als we lokale features zoeken.

Low level features hakken een plaatje in kleine patches op. Die patches variëren in afmetingen van 3x3 pixels tot 24x24 pixels. Elke patch wordt dan doorzocht op aanwezigheid van typische structuren of sterke grijswaarde veranderingen. De sterkste patches worden keypoints genoemd, die worden teruggegeven.

Voor deze opdracht gebruiken we het openCV algoritme ORB. ORB detecteert grote veranderen in grijswaardes in een plaatje, doet daar wat filters over om alleen de meest zinnige veranderingen over te houden.

In de les ga ik zo uitleggen wat dat is, voor nu mag je eerst implementeren en zien wat er gebeurt:

1. Begin in een schoon pythonscript en kopieer weer het basisscript van bovenaan deze opdracht
2. We veranderen het plaatje in grijswaarden omdat de computer toch weinig op heeft met kleuren:

   ```
   gray = cv.cvtColor(img,cv.COLOR_BGR2GRAY)
   ```

3. Nu gaan we de ORB feature detector initialiseren met **`ORB_create`** en daarna detecteren we met **`orb.detect`** alle keypoints in ons autosnelweg plaatje.

   ```
   orb = cv.ORB_create()
   kp = orb.detect(gray,None)
   ```

4. En dat was het. Laten we eens kijken wat het algoritme gevonden heeft, door de keypoints features in het plaatje te tekenen:

   ```
   resultaat = cv.drawKeypoints(img, kp, None, color=kleur_groen,)
   cv.imshow('resultaat',resultaat)
   ```

5. Run het script en bekijk het resultaat. Zou dit genoeg is voor een computer om objecten op te traceren en detecteren? Licht je antwoord toe.