# Structured data acquisition

* Think about technical requirements, e.g. acquisition angles, distances
* the dataset/problem should be challenging enough to show the importance of the various intricacies that are necessary to train a classifier for a challenging problem.
* Move the camera wrt the objects, change angle, distance, etc.

# ML features

• Technically, it is possible to use every pixel in the image as a feature, however this is very unnecessary and inefficient since most of the pixels do not contain relevant information to the algorithm. To make the algorithm more efficient, only meaningful features should be used.

* Complete images rescaled to 50x50x3
* Integrated results of morphological operators, e.g. using vertical and horizontal structuring elements
* Color parameters
  + Hue
  + Saturation
  + Vibrancy(?)
* Color ratios:
  + White surface wrt total object surface
  + Ratio of red area to overall area
  + Ratio of red to blue area
* Contour derived features:
  + Perimeter
  + Area
  + Aspect ratio
  + Extent
  + Nr of vertices in polygon approximation
  + Nr of convexity defects
  + Area of convexity defects
  + average of biggest convexity defect areas, etc.
  + Defect depths, or maximum hull-to-contour distance perpendicular to convex hull
  + Solidity: ratio of contour enclosed area to convex hull area
  + Equivalent diameter: diameter of the circle whose area is the same as the contour area
  + Isoperimetric quotient
  + Shape descriptors (like Hu moments, see e.g. [OpenCV Shape Descriptor: Hu Moments Example - PyImageSearch](https://pyimagesearch.com/2014/10/27/opencv-shape-descriptor-hu-moments-example/))
* Parameters from blobs within an object:
  + Nr of such blobs, also nr of contours
  + Some contour derived parameters but now applied to separate blobs
  + Also defined as ‘holes’ in an object
* Key point:
  + ORB
  + FAST
* Bag of Visual Words approach: SIFT to extract feature descriptors, quantized using K-means.

# DL preprocessing (enhancement + segmentation)

* Bij preprocessing bereid je data die verzameld is voor op het gebruik in het netwerk. Denk hierbij aan het verwijderen van foutieve foto’s en ervoor zorgen dat alle foto’s dezelfde resolutie hebben.
* Bv. converteren van een BGR(RGB) naar HSV-afbeelding. Alle afbeeldingen hebben dezelfde egale achtergrond. Bij een HSV-afbeelding kan gemakkelijk de achtergrond uit alle afbeeldingen gefilterd worden, door op die kleur te filteren.
* Bv. histogram equalization, WB correction, om minder afhankelijk van bv. de witbalans, belichtingstijd ed. te worden
* evt. segmentatie door cropping van RoI
* Afbeeldingen kleiner maken
* Overgaan naar intensiteit plotjes
* Pixelwaardes normalizeren en naar floating point omzetten, evt. Met rescaling laag in TF model
* Ruis onderdukken, b.v. low-pass of erosion

# CNN architecture

* convolutie lagen zijn omringt door normalization en activation lagen om ervoor te zorgen dat de input en outputs van deze lagen niet exploderen in enorm hoge waardes
* De dropout layer in elk blok voorkomt overfitting door tijdens training willekeurig bepaalde verbindingen tussen lagen weg te laten vallen
* Wat opvallend is dat de hoeveelheid filters na blok 1 hoger worden bij elke laag. Na verder onderzoek blijkt dit een goeie design practice te zijn dus hebben we deze overgenomen in ons uiteindelijke netwerk.???
* Twee dense layers bij de output?
* Vgg net: Notice that the numbers of convolution kernels in each CNN layer are increasing larger from first layer to following back layers, this can increase the efficiency of extracting features from images, this method can improve the fitting capacity and the prediction precision of the model.
* The purpose of batch normalization is to normalize the output values of each layer to make them spread evenly. This has the effect of stabilizing the learning process and significantly reducing the number of training epochs required to train deep networks

# Learning curves

* Very fast drop of error in training fase may be an early indicator of overfitting

# Feature maps

* How to create
* How to iterprete

# Bias vs Variance

Algorithm bias, dit gebeurt wanneer er een probleem in de opgezette algoritme zit.

- Sample Bias, dit gebeurt wanneer er een probleem met de data is. Denk hierbij aan niet genoeg (representatieve) data.

- Prejudice Bias, dit gebeurt wanneer er *real world* stereotypes of aannames gemaakt worden.

- Measurement bias, dit gebeurt wanneer er al fouten zijn gemaakt bij het verkrijgen van de data.

- Exclusion Bias, dit gebeurt er wanneer er belangrijke data niet wordt meegenomen in het model.