## Vision

### OpenCV

OpenCV er et open source projekt, oprindeligt udviklet af Intel, til objekt detektering, ansigtsgenkendelse, motion tracking og en lang række andre ting. [[1]](#footnote-1) I dette projekt er det blevet brugt til at detektere duplo klodser ved hjælp af et kamera.

### QRTools

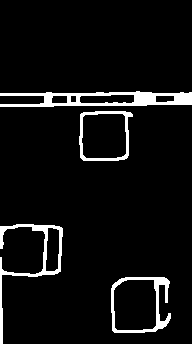
Til at oprette, og aflæse QR koder er QRTools[[2]](#footnote-2) til python blevet anvendt. QRtools er et bibliotek der er sammensat af projekterne qrencode[[3]](#footnote-3) som anvendes til at oprette QR mønstre, og zbar[[4]](#footnote-4) som bruges til at aflæse koderne med. QRTools er valgt fordi det er let at bruge, og anvendelsesmulighederne er mange. Et problem med det var dog at det ikke umiddelbart er kompatibelt med ip kameraer, som er den type kamera der er anvendt i dette projekt. Dette er blevet løst ved kontinuerligt at gemme et billede lokalt ved hjælp af OpenCV, og så scanne dette billede med QRTools. I QR koden er der gemt en tekststreng som, i koden, bliver anvendt til at vælge hvilket mønster der skal bygges. Et problem opstod da vi skulle dekode QR koden tilbage til en tekststreng, fordi strengen som QRTools returnererede var formateret i udefineret format. Problemet var at man ikke kunne sammenligne den streng med en almindelig streng i python. Løsningen blev at sætte default encoding i python scriptet til UTF-8, og derefter fungererede scriptet efter hensigten.

### Detektering af klodser

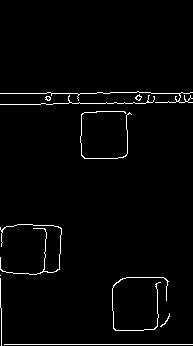
Til at detektere klodserne anvendes kameraet som er monteret ovenover crustcrawler robotten.   
For at klods detektionen kan fungere så optimalt som muligt, har det været nødvendigt at omskrive den udleverede kode[[5]](#footnote-5) til en mere stabil løsning. Størstedelen af genkendelseskoden er indeholdt i funktionen detect brick. Denne funktion tager billedet der skal undersøges, samt øvre og nedre grænseværdier for farven på klodsen angivet i BGR værdier som parametre. Koden for denne funktion vil blive gennemgået i detaljer her.



Først beskæres billedet så unødig genskind fra kanter af bordet fjernes. Dette gøres også for at få arbejdsområdet for robotten, så andet unødigt støj ligeledes bortsorteres.   
Efter dette anvendes medianBlur til at sløre billedet en smule, for at udjævne billedet. Derefter laves det om til gråtoner, da det er nemmere at detektere kanter på et gråtone billede. Canny edge detection anvendes for at trække kanterne i billedet længere frem inden threshold funktionen transformerer billedet om til et binært billede[[6]](#footnote-6) således at kanterne bliver hvide, og resten er sort. Efter dette anvendes dilation til at forstørre kanterne så det bliver nemmere at finde konturerne i billedet efterfølgende. Næstefter anvendes morphologyEx til yderligere at tydeliggøre kanterne med, og til sidst anvendes findContours til at finde konturerne i billedet.  
I for løkken traverseres alle fundne konturer igennem, og der approksimeres en konturform og størrelse samt arealet af konturen. Der valideres så på om arealet holder sig inde for nogen grænseværdier, og hvis dette er tilfældet detekteres centerpunktet for konturen. BoundingRect bliver brugt til at finde den, for klodsen, omkringliggende firkantkontur, og billedet beskæres for at sortere andet fra inden farven detekteres. Det beskårede billede bruges så til at finde en middelværdi for farven på klodsen. Den fundne farve bliver så indsat i et numpy array, og der verificeres at arrayet ikke består kun af 0 værdier. Såfremt arrayet består af 0 værdier betyder det at ingen farve er funden, og den bliver derfor sorteret fra. Er der fundet en farve gemmes denne farve i en tuple sammen med centerkoordinatet, og denne tuple returneres fra funktionen.



Figur - resultat efter morphologyEx, og thresholding



Figur - resultat efter canny edge detection og beskæring af billedet

1. https://en.wikipedia.org/wiki/OpenCV [↑](#footnote-ref-1)
2. https://launchpad.net/qr-tools [↑](#footnote-ref-2)
3. https://github.com/Arachnid/pyqrencode/tree/master [↑](#footnote-ref-3)
4. http://zbar.sourceforge.net/ [↑](#footnote-ref-4)
5. Se au\_opencv\_example.py [↑](#footnote-ref-5)
6. Et binært billede er et billede hvor en pixel kun kan antage to værdier, typisk sort/hvid [↑](#footnote-ref-6)