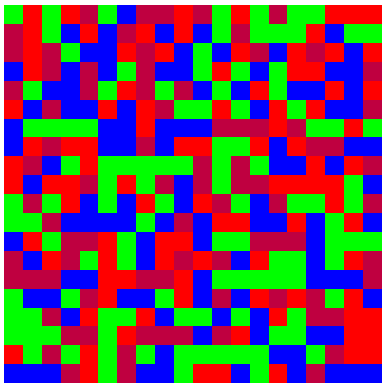
1 Monte Carlo Lokalisering

Dette afsnit vil komme ind på, hvordan en autonom robot kan implementere en Monte Carlo Algoritme for således at kunne lokalisere sig selv på et kendt kort.

Der tages udgangspunkt i en autonom robot, der har fire hjul og mindst to elektriskemotorer med enkoder. Elektriskemotorer med enkoder afgiver et elektrisk signal n antal gange per total rotation af motorens akse. Dette kan eksempelvis være implementeret ved brug af en hall sensor, der måler på en magnet eller en optisk sensor, som måler på et optisk gitter der følger aksen. Således er robotten i stand til at udregne, hvor langt den har bevæget sig i en given retning. Denne robot kunne f.eks. være en reklame robot, den kører rundt i et givent område og har en bakke med chokolade som folk kan tage af. Dette område kunne være en banegård eller en butik. Robotten har her til en sensor der er placeret på undersiden, denne sensoren måler farven af hvad robotten bevæger sig på. Sensoren måler et resultat på formen (r, g, b), hvor $(0, 0, 0)$ vil være sort og $(255, 0, 0)$ ville være en rød farve. De forskelige farver der måles kunne være forskelige nuancer eller farver fliser [**DR**].

Med et udgangspunk i overstående robot der implementerer et Dead-Reckoning algoritme, kan vi lokalisere robotten på et kendt område ved brug af Monte Carlo Lokalisering. Det kendte område kan betragtes som et kort, i dette konkrete eksempel er kortet kvadratisk af 20×20 meter og er bestående af fliser med forskellige farver, disse farver er henholdsvis rød, grøn, blå og lilla, som illustreret på nedenstående figur.



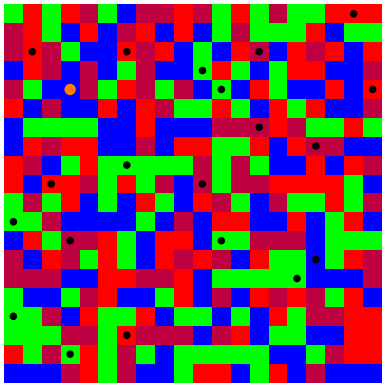
Altså vi har et kendt kort hvor der er 400 fliser af forskelige farver hvilke også fungerer som punkter. Dette er repræsenteret i vores udfaldsrum U. Vores udfaldsrums elementer er punkter på formen $u_i = (x, y)$ og $rgb(u_i)$ er flisens farven på formen RGB, altså hvis vi har en rød flise på punktet $(1, 2)$ vil $rgb((1, 2)) = (255, 0, 0)$. Dette udfaldsrum er en del af et sandsynlighedsfelt, hvilket betyder at robotten har en sandsynlighed for at befinde sig på et givent punkt. Ud fra dette sandsynlighedsfelt kan vi definere en stokastisk variabel X som kan tage hvilket som helst punkt i udfaldsrummet U, hvor x_i er et konkrete punkt. Algoritmen består i et løbende antal iterationer, hvor under hver iteration bliver der simuleret et n antal konkrete stokastisk variabler x_i. Disse konkrete punkter bliver også kaldet for partikler, hvilket bliver tydeligt på de kommende illustreringer. Desuden er sandsynlighedsfunktionen P afhængig af om der er tale om den første iteration af

algoritmen eller om det er løbende iterationer. For første iteration er sandsynlighedsfunktionen den samme som ved et symmetrisk sandsynlighedsfelt, altså der er tale om en jævn fordeling af partiklerne, men de efterfølgende iterationer er afhængig af kun og kun den forgående iterations partikler. sandsynlighedsfunktionen er afhængig på følgende måde $P(x_i|x_{i-1})$ hvor x_{i-1} er alle de konkrete

Nedenstående trin er derfor involverede i at lokalisere robotten på kortet.

1. n tilfældige konkrete stokastisk variabler x_i bestemmes.
2. Robotten laver en måling m på formen (r, g, b) på dens nuværende position
3. Målingen m sammenlignes nu med x_i på følgende måde $|m - x_i|$
4. Et nyt sandsynlighedsfelt λ hvor $P(x_i) = \frac{|m-x_i|}{s}$ hvor s er summen af alle fejl fra forgående trin og U er de forgående konkrete elementer x_i. Dette vil betyde at der er større sandsynlighed for de punkter med en stor fejl relativt til dem med en lav fejl
5. Der bestemmes nu λ_n tilfældige punkter fra λ, hvor $\lambda_n < n$, og fjernes fra λ_U
6. Robotten bevæger sig nu en flise i hvilken som helst retning, hvor robotten stadig vil være inden for fliserne. Alle resterende punkter i λ flyttes i samme retning og længde
7. Der bestemmes nu λ_n, altså samme antal som blev fjernet, punkterne fra et sandsynlighedsfelt hvor $P(x_i) = 1 - \frac{|m-x_i|}{s}$, altså så punkter med en lav fejl har en højere sandsynlighed. En af de omliggende fliser fra dette punkt som robotten kunne have bevæget sig over vælges og tilføjes til udfaldsrummet λ_u. Derefter gentages de forgående tre trin det antal gange robotten bevæger sig.

Hvis man fulgte overstående trin, vil man finde at man starter ud med en masse punkter spredt ud over hele kortet, men som processen gentages bliver der dannet en koncentration af punkter hvor robotten faktisk befinder sig. Nedenstående figurer illustrerer dette, hvor de sorte punkter er mulige positioner for robotten og det orange punkt er robotten.



I nedenstående sektion, vil jeg komme ind på en teoretisk implementation af dette i Python.