

PROSJEKT - HØSTEN 2019

| | |
|-----------------------|---|
| Prosjekt- oppgaven | LEGO Mindstorms og MATLAB; anvendt matematikk/ fysikk og programmering i skjønn forening |
|-----------------------|---|

| | | | |
|-----------------------|------|---------------|---|
| Gruppenummer | 19XX | | |
| Gruppens medlemmer | Navn | Studentnummer | Bilde |
| | Per | 6378983 |  |
| | Pål | 6378984 |  |
| | Eva | 6378985 |  |
| | Anne | 6378986 |  |

Innhold

| | |
|--|-------------|
| Innhold | ii |
| Sammendrag | viii |
| 1 Numerisk integrasjon (utført av hele gruppen) | 1 |
| 1.1 Problemstilling | 1 |
| 1.2 Forslag til løsning | 1 |
| 1.3 Resultat | 1 |
| 1.3.1 Integrasjon av en konstant | 1 |
| 1.3.2 Integrasjon av et sinussignal | 1 |
| 2 Filtrering (utført av hele gruppen) | 2 |
| 2.1 Problemstilling | 2 |
| 2.2 FIR-filter | 2 |
| 2.2.1 Kode | 2 |
| 2.2.2 Resultat | 2 |

INNHOLD

| | | |
|----------|---|----------|
| 2.3 | IIR-filter | 2 |
| 2.3.1 | Kode | 2 |
| 2.3.2 | Resultat | 2 |
| 2.4 | Sammenligning | 2 |
| 3 | Numerisk derivasjon (utført av hele gruppen) | 3 |
| 3.1 | Problemstilling | 3 |
| 3.2 | Forslag til løsning | 3 |
| 3.3 | Resultat | 3 |
| 3.3.1 | Derivasjon av et lineært signal | 3 |
| 3.3.2 | Derivasjon av et sinussignal | 3 |
| 4 | Manuell kjøring av Lego-robot (utført av hele gruppen) | 4 |
| 4.1 | Problemstilling | 4 |
| 4.2 | Forslag til løsning | 4 |
| 4.3 | Resultat | 4 |
| 5 | Automatisk kjøring av Legorobot (utført av Anne og Kari) | 5 |
| 5.1 | Problemstilling | 5 |
| 5.2 | Forslag til løsning P-regulator | 5 |
| 5.3 | Resultat | 5 |
| 5.4 | Forslag til løsning PI-regulator | 5 |

INNHOLD

| | | |
|----------|--|-----------|
| 5.5 | Resultat | 5 |
| 5.6 | Sammenligning | 5 |
| 6 | Turtallsregulator, utført av Anne og Pål | 6 |
| 6.1 | Problemstilling | 6 |
| 6.2 | Forslag til løsning | 7 |
| 6.2.1 | LEGO-konstruksjon | 8 |
| 6.2.2 | Kode for beregning av vinkelhastighet | 9 |
| 6.2.3 | Testing uten turtallsregulator | 11 |
| 6.2.4 | Sammenheng mellom pådrag og vinkelhastighet | 13 |
| 6.2.5 | Kode for turtallsregulator | 16 |
| 6.3 | Resultat | 19 |
| 6.3.1 | P-regulator | 19 |
| 6.3.2 | PI-regulator | 21 |
| 6.3.3 | Integratorbegrensing | 23 |
| 6.3.4 | PID-regulator | 27 |
| 6.3.5 | Effekten av å kople inn turtallsregulatoren | 28 |
| 6.4 | Turtallsregulator som funksjon | 31 |
| 7 | Litt om referering til rapportelementer | 35 |
| 7.1 | <i>Tagging</i> av rapportelement, <code>\label{ }</code> | 35 |

INNHOLD

| | |
|--|--------|
| 7.2 Tagging av kode, <code>\label{}</code> | 37 |
| 7.3 Bruk av <code>\ref{}</code> , <code>\eqref{}</code> og <code>\pageref{}</code> | 38 |
| 7.4 Beskrivelse av litteratur | 39 |
| 7.4.1 <code>@Book</code> | 40 |
| 7.4.2 <code>@TechReport</code> | 41 |
| 7.4.3 <code>@Misc</code> | 42 |
| 7.4.4 <code>@Article</code> | 43 |
| 7.4.5 <code>@InProceedings</code> | 45 |
| 7.4.6 <code>@PhDThesis</code> og <code>@MastersThesis</code> | 47 |
| 7.5 Referering til litteratur, <code>\cite{}</code> | 48 |
| 8 Litt om ligninger | 49 |
| 8.1 Bruk av <code>equation</code> -miljøet | 49 |
| 8.2 Bruk av <code>align</code> -miljøet | 49 |
| 8.3 Bruk av matematikkmodus i rapporttekst, <code>\$x\$</code> | 50 |
| 8.4 Bruk av vanlig tekst i ligninger, <code>\mathrm{}</code> | 51 |
| 8.5 Bruk av <code>\hspace*</code> | 51 |
| 9 Litt om figurer | 52 |
| 9.1 Bruk av <code>figure</code> -miljøet | 52 |

INNHOLD

| | |
|--|-----------|
| 9.2 MATLAB-figurer | 53 |
| 9.2.1 Format og størrelse | 53 |
| 9.2.2 Funksjonen <code>SaveMyFigure.m</code> | 55 |
| 9.2.3 Effekten av vindusstørrelse <i>før</i> lagring | 56 |
| 9.2.4 Inkludering av L ^A T _E X-tekst i MATLAB-figurene | 58 |
| 10 Litt om tabeller | 60 |
| 10.1 Bruk av <code>table</code> - og <code>tabular</code> -miljøene | 60 |
| 11 Litt om punktlister | 63 |
| 11.1 Bruk av <code>\itemize</code> | 63 |
| 11.2 Bruk av <code>\enumerate</code> | 65 |
| 11.2.1 Tallbasert liste, standardversjonen | 65 |
| 11.2.2 Bostavbasert liste | 65 |
| 11.2.3 Romertallbasert liste | 66 |
| 12 Litt om inkludering av MATLAB-kode | 67 |
| 12.1 Pakkene <code>listings</code> og <code>mcode</code> | 67 |
| 12.1.1 Dynamisk med <code>\lstinputlisting</code> -kommandoen | 67 |
| 12.1.2 Statisk med <code>\lstlisting</code> -kommandoen | 69 |
| 12.1.3 Pakken <code>figurepath</code> | 70 |

INNHOLD

| | |
|--|-----------|
| 13 Litt om Overleaf | 73 |
| 14 Konklusjon | 86 |
| Bibliografi | 87 |
| Vedlegg | 87 |
| A Timelister | 88 |
| B Programlisting prosjekt 04 | 89 |
| B.1 P04_F4_MathCalculations.m | 89 |
| B.2 P04_F5_CalculateAndSetMotorPower.m | 89 |
| C Programlisting prosjekt 05 | 90 |
| C.1 P05_F4_MathCalculations.m | 90 |
| C.2 P05_F5_CalculateAndSetMotorPower.m | 90 |

Sammendrag

Det er i hovedsak 4 hensikter med dette dokumentet og de tilhørende filene i mappen `Rapport_LaTeX`:

1. Det fungerer som et skall for å skrive ING100-rapporten i \LaTeX . Delkapitlene i kapittel 1 til 5 er bare et forslag til inndeling.
2. Det gir et eksempel på hvordan et prosjekt kan dokumenteres i kapittel 6. Grunnen til at dette prosjektet med resultater serveres deg i sin helhet er at du har stor nytte av å anvende tuttallsregulering i prosjektene du gjennomfører. Stoffet kan oppleves som vanskelig, og derfor fremstår dokumentasjonen av prosjektet som relativt omfattende. Poenget er at du skal dokumentere på lignende måte dine egne prosjekt i ING100. Tenk derfor etter mens du leser gjennom kapittel 6 om noe er uklart, eller om du opplever det som overforklart. ING100-rapporten din skal nemlig skrives for en person som har samme bakgrunn som deg selv, og det er viktig at du tenker gjennom hvordan du formulerer deg og på hvilket nivå du legger forklaringene.
3. Det fungerer som et oppslagshefte for tips til \LaTeX . Gjelder kapittel 7 til 12.
4. Det gir et krasjkurs i Overleaf. Gjelder kapittel 13.

Underveis i kapittelet brukes slike oransje kommentarbokser til å fremheve noen poenger om rapportskriving.

Kapittel 1

Numerisk integrasjon (utført av hele gruppen)

1.1 Problemstilling

1.2 Forslag til løsning

1.3 Resultat

1.3.1 Integrasjon av en konstant

1.3.2 Integrasjon av et sinussignal

Kapittel 2

Filtrering (utført av hele gruppen)

2.1 Problemstilling

2.2 FIR-filter

2.2.1 Kode

2.2.2 Resultat

2.3 IIR-filter

2.3.1 Kode

2.3.2 Resultat

2.4 Sammenligning

Kapittel 3

Numerisk derivasjon (utført av hele gruppen)

3.1 Problemstilling

3.2 Forslag til løsning

3.3 Resultat

3.3.1 Derivasjon av et lineært signal

3.3.2 Derivasjon av et sinussignal

Kapittel 4

Manuell kjøring av Lego-robot (utført av hele gruppen)

4.1 Problemstilling

4.2 Forslag til løsning

4.3 Resultat

Kapittel 5

Automatisk kjøring av Legorobot (utført av Anne og Kari)

5.1 Problemstilling

5.2 Forslag til løsning P-regulator

5.3 Resultat

5.4 Forslag til løsning PI-regulator

5.5 Resultat

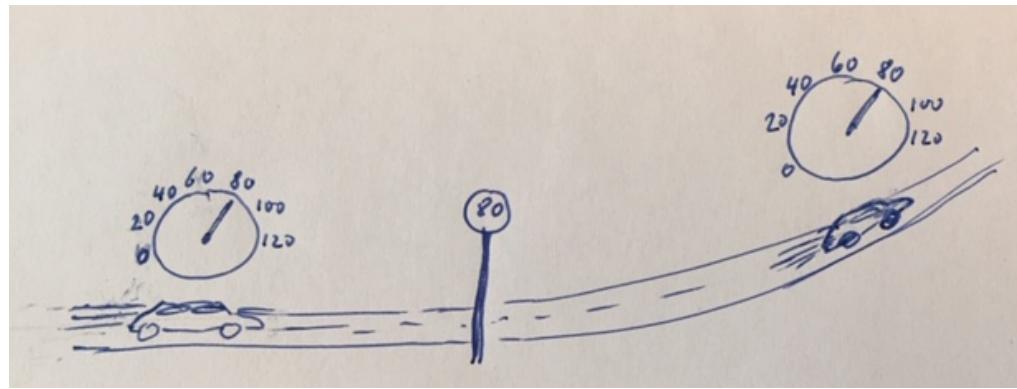
5.6 Sammenligning

Kapittel 6

Turtallsregulator, utført av Anne og Pål

6.1 Problemstilling

I dette prosjektet har målet vært å lage en turtallsregulator for LEGO-motorene. Funksjonen til en turtallsregulator kan sammenlignes med en vanlig (ikke adaptiv) *cruise controller* i en bil, hvor hastigheten til bilen opprettholdes selv om bilen møter økt motstand i form av en motbakke, se illustrasjonen i figur 6.1.



Figur 6.1: En bil med *cruise controller* aktivert vil automatisk holde samme hastighet, uansett om bilen kjører rett frem eller i en motbakke.

Bruk alltid innledende tekst mellom overskrift og eventuelle figurer.

Engelske begrep skrives i *kursiv* font.

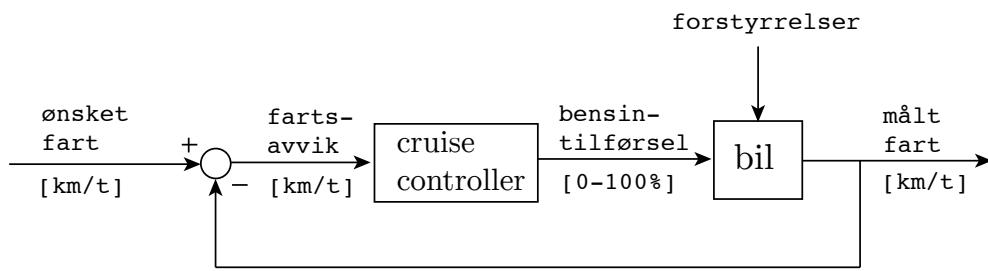
Referer alltid til figur før den kommer i teksten.

Enkel skisse/figur som illustrerer problemstillingen på en gjenkjennende måte. Et tegneprogram som kan være nyttig å lære seg er `inkscape` som lager figurer i vektorgrafikk (bedre enn punktgrafikk).

6.2 Forslag til løsning

For å få dette til blir bilens målte hastighet sammenlignet med ønsket hastighet (som ofte er det samme for fartsgrensen), hvor forskjellen fartsavvik (se figur 6.2) brukes av regulatoren til å automatisk justere bensintilførselen (pådraget).

Variabelnavn som brukes i figur beskrives i teksten, og fonten er den samme slik at leseren ser sammenhengen.



Figur 6.2: Blokkskjema som viser prinsippet bak en *cruise controller* i en bil. Forstyrrelsene som påvirker bilen vil for eksempel være motbakke, nedoverbakke, medvind og motvind.

Dersom avviket mellom ønsket og målt fart f.eks. er postivt (det vil si at ønsket fart er større enn målt fart), vil regulatoren øke pådraget (bensintilførselen) helt til målt hastighet er lik ønsket hastighet (og motsatt dersom avviket er negativt)¹.

Prosjektet som er beskrevet/dokumentert i dette kapittelet har gått ut på å lage tilsvarene funksjonalitet for turtallsregulering av LEGO-motorene. Dette kommer spesielt til nytte i de prosjektene hvor to motorer brukes til å kjøre roboten fremover, og hvor samme pådrag blir gitt til begge motorene. Siden motorene ikke vil møte identisk motstand/friksjon langs veien, vil man kunne oppleve at roboten utilsiktet svinger dersom vi ikke benytter en turtallsregulator.

Eksempel på at noe er skrevet i fortid siden arbeidet er allerede utført.

6.2 Forslag til løsning

I dette kapittelet presenterer vi vårt forslag til løsning av prosjektet. Dette inkluderer en LEGO-konstruksjon som vi bygget for at LEGO-motoren skulle kunne utsettes for relativt lik friksjon i forsøkene. I tillegg presenteres koden for å beregne vinkelhastighet og koden for å implementere PID-regulatoren.

Ved å bruke et par setninger i begynnelsen av et kapittel hvor du beskriver hva kapittelet handler om, gjør du leseren forberedt på innholdet.

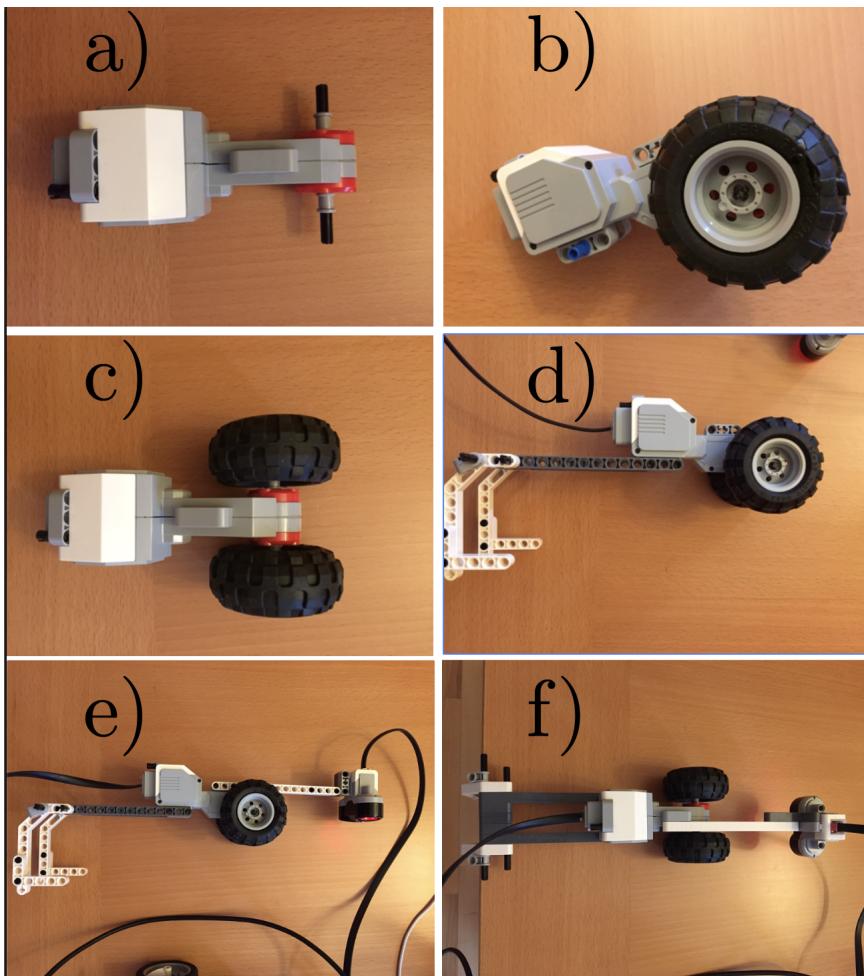
¹Ved kjøring uten *cruise control* er det hjernen vår som utfører denne reguleringsfunksjonen hvor høyrefoten justerer på gasspedalen (eller bremsen) for å oppnå ønsket hastighet.

6.2 Forslag til løsning

6.2.1 LEGO-konstruksjon

For å kunne teste ut turtallsreguleringen i praksis, lagde vi følgende LEGO-konstruksjon, se figur 6.3.

Setninger som begynner med "For å..." gir hensikt og motivasjon.



Figur 6.3: Byggebeskrivelse for repeterbart å kunne tilføre noenlunde identisk friksjon på motoren. a) Aksling med avstandshylser montert. b) og c): Hjul montert på begge sider av motoren. d): Konstruksjon for å holde motoren igjen på bordkanten. e): Ultralydsensor montert foran motoren for å registrere om konstruksjonen løftes opp fra bordet (friksjon fjernes). f) Hele konstruksjonen sett ovenfra.

Figuren beskrives i caption ved å referere til delfigurer.

6.2 Forslag til løsning

Gummihjulene som er montert på motoren gjør det mulig å utsette motoren for friksjon når hele konstruksjonen hviler mot et bord. Siden vekten av motoren og LEGO-delene ikke endres, er det mulig å få relativt like friksjonsbetingelser under forskjellige eksperimenter.

Ved å løfte på selve motoren mens den henger på bordkanten vil friksjonen forsvinnne. For å kunne registrere dette i datasettet, benyttes ultralydsensoren (som måler avstand). Dersom motoren løftes og kjøres uten friksjon, vil avstand til bordet bli stor. Når konstruksjonen slippes ned på bordet, vil avstand til bordet bli liten.

Rapporten skal i hovedsak presentere løsningen som til slutt fungerte. Dersom flere ting har vært utprøvd, er det viktig å ikke lage historiefortelling som: "....først så prøvde vi trykkbryteren..... Deretter prøvde vi ultralydsensoren....", men heller først presentere det som virket, og deretter nevne de tingene som ikke virket og en forklaring på hvorfor de ikke virket.

Legg merke til at samme informasjon også finnes i figurteksten til figur 6.3, men at dette ikke oppleves som unødvendig repetisjon.

Vi prøvde også å bruke trykkbryteren for å registrere hvorvidt konstruksjonen hviler på bordet, men denne krevde litt for mye kraft for å bli trykket inn, og dermed oppsto det av og til feilmålinger.

6.2.2 Kode for beregning av vinkelhastighet

I dette delkapittelet presenterer vi først koden for å styre motorpådraget, og deretter koden for å beregne vinkelhastigheten og turtall ut fra avlest vinkelposisjon på motoren.

For å styre en LEGO-motor (som ikke er turtallsregulert) settes pådraget til en verdi på mellom -100 og 100. Dette tallet kan tolkes til å ha enheten [%], hvor positivt verdi roterer "fremover", mens negativt tall roterer "bakover". I uttestingsfasen av turtallsregulatoren benyttet vi *potensiometeret* (se figur 6.4) som pådrag til motoren. Dette fordi vi på denne måten kunne holde pådraget helt konstant/fast over en tidsperiode, men samtidig at det var enkelt å justere nivået på pådraget i samme eksperiment (som et sprang)². Kodeutdraget under viser hvordan avlest potensiometerverdi PotMeter(k) (linje 1) benyttes direkte som pådragsverdi til motor A, PowerA(k) (linje 2). De to siste kommandolinjene starter motor A.

Figur 6.4 kommer på toppen av neste side siden vi på denne figuren bruker [ht] istedenfor [H], se L^AT_EX-koden i chapter6.tex.

Kode 6.1: Kode for å bestemme og sette pådraget til motor A (Mac-versjon)

```
1 PotMeter(k) = -skalering*axis(joystick,4);  
2 PowerA(k) = PotMeter(k);  
3 motorA.Speed = PowerA(k);  
4 start(motorA);
```

I beskrivelsen av koden er det viktig at variabelnavn skrives i samme font (gir gjenkjennning) og at det hjelper leseren der som du angir linjenummer.

Prinsippet bak å beregne vinkelhastigheten (i enheten [grader/s]) er å først avlese

²Alternativt kunne vi benyttet selve styrestikken, men denne er det vanskelig å holde i en konstant posisjon.

Hele kodeutdraget består av kodelinjer hentet fra filene P08_F3_... og P08_F5_... og er presentert på en kompakt måte for leseren.

6.2 Forslag til løsning



Figur 6.4: Potensiometeret indikert med hvit ring benyttes som pådrag til motoren.

vinkelposisjonen til motor A (i enheten [grader]) og deretter deriverer denne ved hjelp av derivasjonsfunksjonen fra kapittel 3. Siden vinkelposisjonsmålinger er befengt med støy (vil gi ubruklig resultat etter derivering), så blir disse først filterert. Kodeutdrag 6.2 viser nødvendig kode,

Kode 6.2: Kode for beregning av vinkelhastighet til motor A.

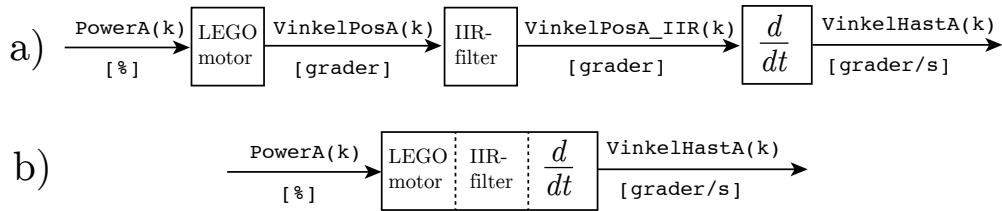
```
5 VinkelPosA(k) = double(motorA.readRotation);
6 alfa1 = 0.3;
7 VinkelPosA_IIR(k) = ...
    IIR_filter(VinkelPosA_IIR(k-1), VinkelPosA(k), alfa1);
8 VinkelHastA(k-1) = Derivation(VinkelPosA_IIR(k-1:k), Ts(k-1));
```

hvor

- `VinkelPosA(k)` i linje 5 er avlest vinkelposisjon.
- Verdien på `alfa1=0.3` i linje 6 ble funnet etter endel prøving og feiling.
- `VinkelPosA_IIR(k)` i linje 7 er filtrert vinkelposisjon.
- `VinkelHastA(k-1)` i linje 8 er beregnet vinkelhastighet.

6.2 Forslag til løsning

Oppsummert kan det som skjer i kodeutdragene 6.1 og 6.2 presenteres som blokkskjemaet i delfigur a) figur 6.5. Delfigur b) er en komprimert versjon som vi gjenbruker senere når vi presentererer turtallsregulatoren.



Figur 6.5: a): Blokkskjema over koden i kodeutdragene 6.1 og 6.2. b): En komprimert fremstilling av blokkskjemaet i a).

6.2.3 Testing uten turtallsregulator

For å finne uten hvordan LEGO-motoren håndterte friksjon uten turtallsregulator, gjennomførte vi et eksperiment hvor konstruksjonen i figur 6.3 ble løftet opp og lagt ned igjen på bordet ved forskjellige pådrag. Et typisk resultat er vist i figur 6.6.

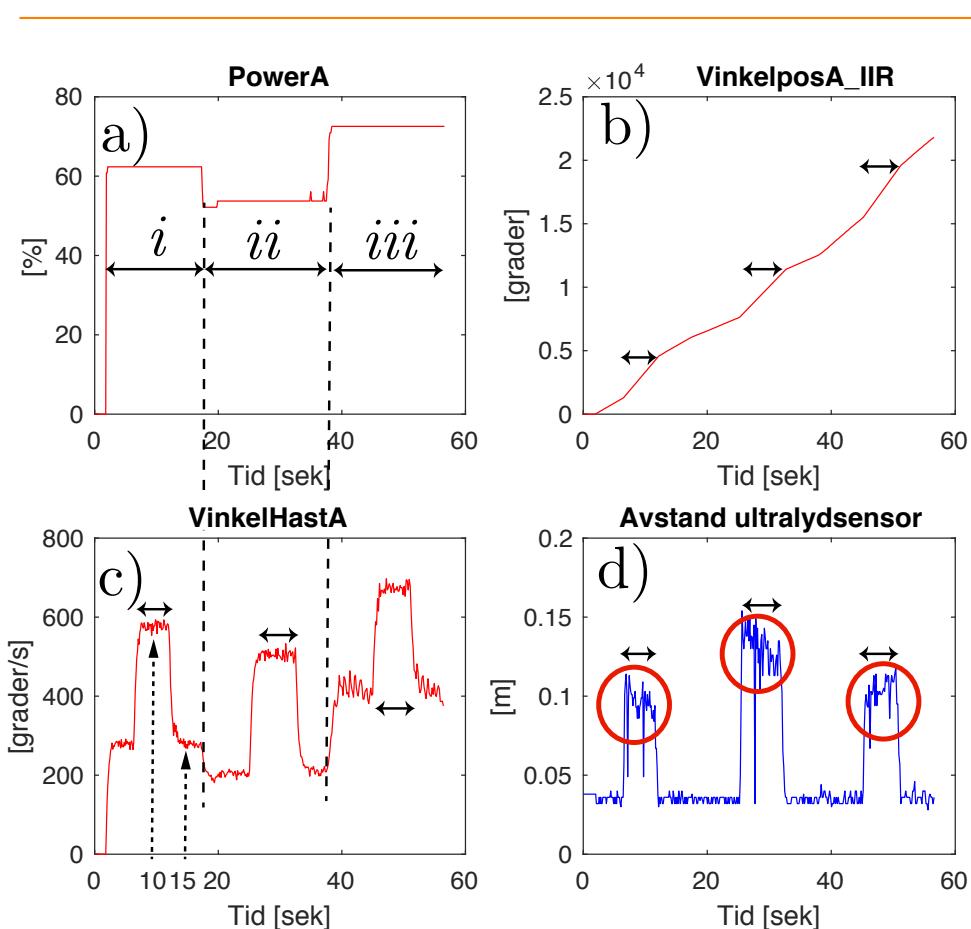
I dette eksperimentet ble det benyttet 3 forskjellige pådrag, markert med *i*, *ii* og *iii* i delfigur a). Midt i tidsperioden for hver av de tre pådragene ble LEGO-konstruksjonen løftet opp, og dette er markert med røde ringer i delfigur d). Når LEGO-konstruksjonen ligger på bordet er avstanden målt med ultralydsensoren ca 4 cm.

Delfigur b) viser målt vinkelposisjon (IIR-filtrert), og siden motoren roterer samme vei gjennom hele eksperimentet, øker denne monotont. Det kan observeres at økning er størst i de periodene hvor LEGO-konstruksjonen er løftet opp fra bordet (markert med piler). I løpet av eksperimentet som varer ett minutt, har motoren roteret ca $2.2 \cdot 10^4$ grader, noe som tilsvarer

$$\frac{22000}{360} \approx 60 \text{ runder} \quad (6.1)$$

I teksten vises det tydelig til detaljer i figuren slik at leseren slipper å lete i figuren etter det som beskrives. Ponenget med ligning (6.1) er bare for å relatere tallverdi-en til noe leseren har et forhold til, og fungerer kanskje mer som *fun fact*.

6.2 Forslag til løsning



Legg merke til at figuren har tydelige og lesbare akseverdier, at aksene har tydelig benevning og titler som beskriver hva som vises, at det inkluderes piler, tekst, sirkler og andre elementer for å tydeliggjør sammenhenger. Dette forenkler også mulighetene til å beskrive resultatene i rapportteksten.

Figur 6.6: Målinger og beregninger fra et eksperiment som viser hvordan friksjon endrer vinkelhastigheten på en LEGO-motor uten turtallsregulator. a): Pådragsverdier til LEGO-motoren, delt inn i perioder markert med *i*, *ii* og *iii* hvor pådraget er forskjellig, men konstant. b): Filtrert vinkelposisjon til motor A. c): Beregnet vinkelhastighet ut fra de filtrerte målingene i b). d): Målinger fra ultralydsensoren hvor periodene indikert med rød ring betyr at LEGO-konstruksjonen i figur 6.3 er løftet opp fra bordet.

Delfigur c) viser den beregnede vinkelhastigheten i [grader/s], og vi ser at vinkelhastigheten halveres når LEGO-konstruksjonen hviler på bordet. Dette kan f.eks. ses ved de lodrette pilene ved tidspunkt $t=10$ og $t = 15$ sekund, hvor vinkelhastigheten først er ca 600 grader/s (tilsvarer at roboten er løftet opp, se delfigur d)), og etter at roboten legges ned på bordet reduseres vinkelhastighet til ca 300 grader/s.

Ved første øyekast i delfigur c) ser du kanskje ikke at vinkelhastigheten halveres, men ved å indikere tidspunkt og bruke piler (eller annet) hjelper du leseren til å forstå resonnementet.

6.2 Forslag til løsning

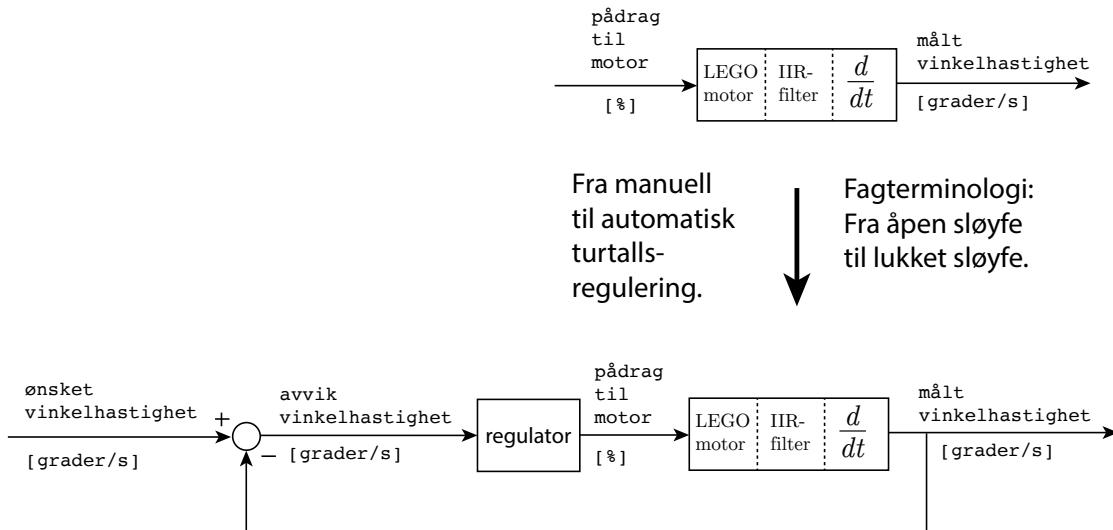
6.2.4 Sammenheng mellom pådrag og vinkelhastighet

Som vist i eksempelet med *cruise controlleren* i figur 6.2, så føres målingen [målt fart] tilbake og sammenlignes med referansen [ønsket fart]. Sammenligningen innebærer en subtraksjon og beregning av *reguleringsavviket* [fartsavvik] som

$$\text{fartsavvik} = \text{ønsket fart} - \text{målt fart} \quad (6.2)$$

En forutsetning for å beregne dette avviket er lik benevningen for referanse og måling, i dette tilfelle [km/t]. På tilsvarende måte må referansen for turtallsregulatoren, [ønsket vinkelhastighet], ha samme benevning som [målt vinkelhastighet], dvs. [grader/s]. Dette er illustrert i figur 6.7 hvor vi har tatt utgangspunkt i det komprimerte blokkskjema fra delfigur b) i figur 6.5 og vist hvordan det prinsipielle blokkskjema for turtallsregulatoren vil se ut.

Ved å ramme inn variabelnavn tydeliggjør du symbolbruken som vises i figuren.



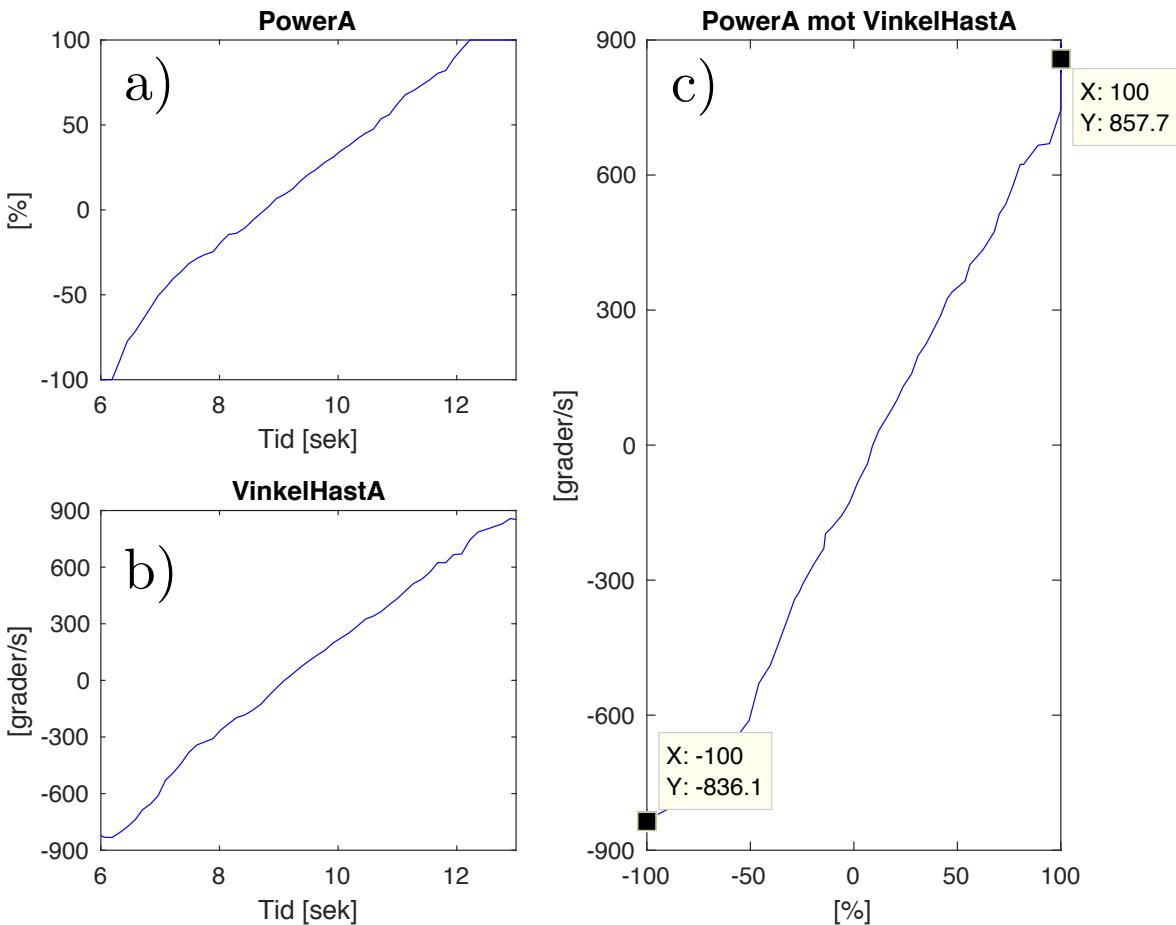
Figur 6.7: Blokkskjema som viser hvordan LEGO-motoren fra delfigur b) i figur 6.5 må kobles opp for å gå fra manuell til automatisk turtallsregulering. Reguleringsteknisk fagterminologi er fra åpen sløyfe til lukket sløyfe.

For at vi skulle kunne spesifisere oppnåelige verdier i [ønsket vinkelhastighet], måtte vi derfor først finne maksimalt oppnåelig vinkelhastighet på LEGO-motoren. Dette ville da være øvre³ grense for referansen. Vi gjennomførte derfor et forsøk

³Og tilsvarende nedre med motsatt fortegn.

6.2 Forslag til løsning

hvor vi rolig økte pådraget til motoren fra -100% til $+100\%$, uten at vi tilførte friksjon på motoren. Resultatet fra dette forsøket er vist i figur 6.8, hvor delfigur a) viser pådraget PowerA som funksjon av tid, delfigur b) viser beregnet vinkelhastighet VinkelHastA , og delfigur c) viser sammenhengen mellom pådrag og vinkelhastighet, dvs. VinkelHastA som funksjon av PowerA .



Figur 6.8: Resultater som viser sammenheng mellom PowerA og beregnet vinkelhastighet VinkelHastA . a): PowerA som funksjon av tid. b): VinkelHastA som funksjon av tid. c): VinkelHastA som funksjon av PowerA .

Som vi kan se fra avlesningen i delfigur c), gir fullt positivt pådrag på motoren en vinkelhastighet på ca 857 [grader/s], men fullt negativt pådrag gir en vinkelhastighet

6.2 Forslag til løsning

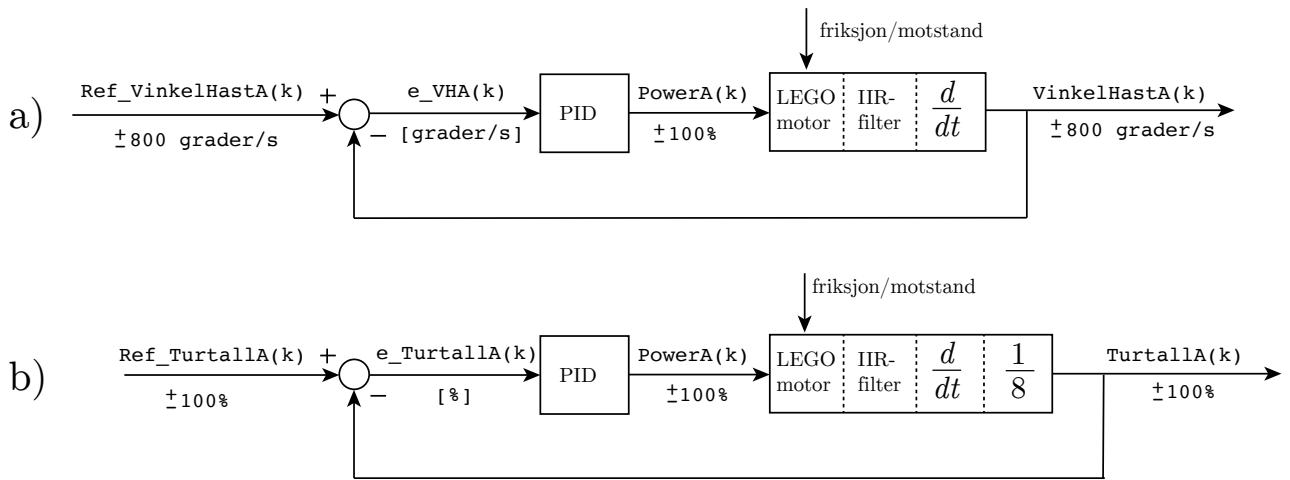
på ca -836 [grader/s]. For å unngå *windup*-problemer i reguleringen (se nedenfor), avrunder vi nedover og bestemmer at maksimal oppnåelig vinkelhastighet (uten friksjon) er ± 800 [grader/s], dvs.

$$\text{PowerA} = \pm 100 [\%]$$



$$\text{VinkelHastA} = \pm 800 \text{ [grader/s]}$$

Dette innebærer at den prinsipielle reguleringssløyfen i figur 6.7 kan presenteres med mer detaljer som vist i delfigur a) i figur 6.9. Legg merke til at referansen⁴ kalles **Ref_VinkelHastA**, siden målingen kalles **VinkelHastA**.



Figur 6.9: a): Blokkskjema over turtallsregulatoren hvor verdiene til referansen **Ref_VinkelHastA** varierer mellom ± 800 grader/s. b): Blokkskjema over turtallsregulatoren hvor verdiene til den relative referanse **Ref_TurtallA** varierer mellom $\pm 100\%$. Legg merke til at friksjon/motstand er tegnet inn som en forstyrrelse på motorblokken.

Det gjør ingenting om figurene dine bruker margene på begge sider, dersom dette trengs for å gjøre de lesbare.

For å forenkle bruken av turtallsregulatoren fant vi ut at det å normalisere vinkelhastigheten til en verdi mellom $\pm 100\%$ gjorde det mye enklere å bytte mellom manuell og automatisk kjøring siden vi da slapp å måtte forholde oss til følgende veldig forskjellige benevninger:

⁴Referansen kalles også *settspunkt* eller *skal-verdi*.

6.2 Forslag til løsning

- pådrag i [%] ved manuell kjøring av motoren og,
- ønsket vinkelhastighet i [grader/s] ved automatisk kjøring

Ved å normalisere vinkelhastigheten med det absolutte variasjonsområdet ± 800 grader/s til et generelt **turtall** med det relative variasjonsrområdet $\pm 100\%$, vil selve *verdien* vi forholder oss til i manuell og automatisk kjøring være identisk. Det betyr altså at variasjonsområdet til pådraget/referansen vil være identisk, og forskjellen vil være hvorvidt turtallsregulatoren er aktiv eller ikke.

Strukturen for denne turtallsregulatoren er vist i delfigur b) i figur 6.9, hvor vi har lagt til en skaleringsblokk på $\frac{1}{8}$ etter derivasjonsrutinen. Siden denne nye måleverdien representerer den generelle størrelsen turtall i [%], har vi endret utgangen og referansen til henholdsvis **TurtallA** og **Ref_TurtallA**. Koden for å beregne TurtallA baserer seg da på kodeutdrag 6.2 i tillegg til følgende kode:

Kode 6.3: Kode for å beregne TurtallA.

```
9 TurtallA(k-1) = 1/8*VinkelHastA(k-1);
```

Koden i dette kapittelet er plassert i `P0X_F4_MathCalculations.m`.

Legg merke til at vi ikke viser koden for ultralydsensoren som registrerer om LEGO-konstruksjonen løftes opp. Dette fordi dette ikke er en viktig del av selve turtallsregulatoren.

6.2.5 Kode for turtallsregulator

I dette delkapittelet blir koden for turtallregulatoren presentert. Den er basert på en standard industriell PID-regulator som matematiske kan uttrykkes som i ligning (6.3),

$$u(t) = u_0 + \underbrace{K_P \cdot e(t)}_{P} + \underbrace{K_I \int_0^t e(t) dt}_{I} + \underbrace{K_D \cdot \frac{d}{dt} e_f(t)}_{D} \quad (6.3)$$

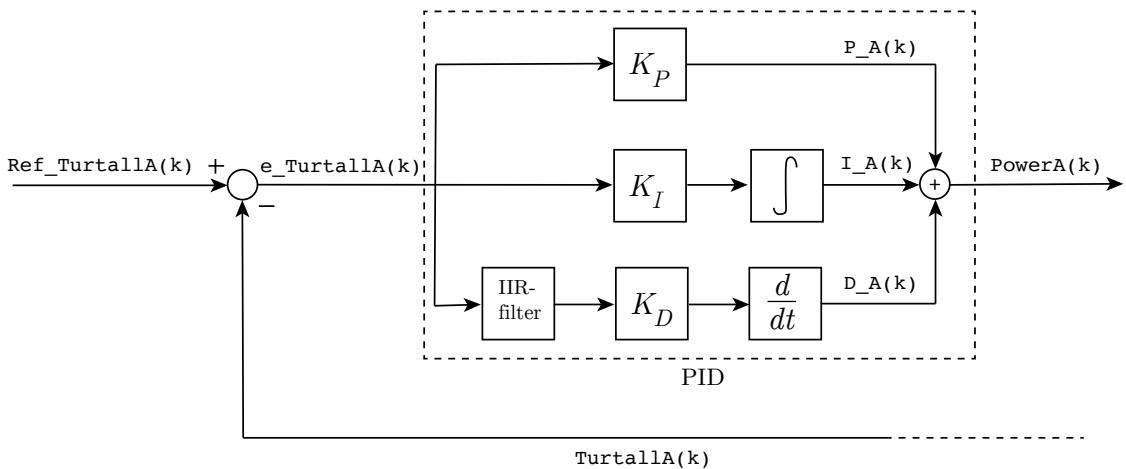
hvor

- u_0 er basispådraget som for turtallsregulatoren er satt lik 0.
- $e(t)$ er reguleringsavviket som er forskjell mellom referansen og målingen
- $e_f(t)$ er filtrert reguleringsavvik
- K_P , K_I og K_D er regulatorparametre.

6.2 Forslag til løsning

- $u(t)$ er det beregnede totalpådraget
- Bokstavene P, I og D står for henholdsvis proporsjonaldel, integraldel og derivatdel.

For å relatere disse generelle variablene og parametrerne til turtallsregulatoren vist som en PID-blokk i figur 6.9b). kan ligning (6.3) skjematisk presenteres som figur 6.10.



Figur 6.10: Blokkskjema over turtallsregulatoren vist som PID-blokk i figur 6.9b) og som er basert på ligning (6.3). Bidragene $P_A(k)$, $I_A(k)$ og $D_A(k)$ tilsvarer bidragene P, I og D i ligning (6.3).

Detaljer om koden for turtallsregulatoren er som følger:

- Reguleringsavviket $e_{TurtallA}(k)$ beregnes først som differansen mellom $Ref_{TurtallA}(k)$ (som settes lik potensiometerposisjonen fra kode 6.1) og beregnet turtall $TurtallA(k-1)$ fra kode 6.3. Dette er vist i kode 6.4.

Kode 6.4: Beregning av reguleringsavviket $e_{TurtallA}(k)$.

```

15  Ref_TurtallA(k) = PotMeter(k)
16  e_TurtallA(k) = Ref_TurtallA(k) - TurtallA(k-1);

```

Legg merke til at vi antar at beregnet turtall i forrige tidsskritt $k-1$ kan brukes til å beregne et avvik i nåværende tidsskritt k .

6.2 Forslag til løsning

- Proporsjonaldelen $P_A(k)$ forsterker reguleringsavviket med parameteren K_P , se kodeutdraget under.

Kode 6.5: Beregning av P-delen for motor A, $P_A(k)$.

```
17     P_A(k) = Kp*e_TurtallA(k);
```

Dersom avviket $e_TurtallA(k)=0$, som jo egentlig er det vi ønsker å oppnå, vil proporsjonaldelen også bli 0.

- Integraldelen $I_A(k)$ integrerer produktet av reguleringsavviket og integrasjonsforsterkningen K_I , se kodeutdraget under.

Kode 6.6: Beregning av I-delen for motor A $I_A(k)$.

```
18     I_A(k) = EulerForward(I_A(k-1), Ki*e_TurtallA(k-1), ...
                           Ts(k-1));
```

Til dette bruker vi integrasjonsfunksjonen `EulerForward` fra kapittel 1.

- Derivatdel $D_A(k)$ deriverer produktet mellom det IIR-filtrerte reguleringsavviket og derivasjonsforsterkningen K_D , se under.

Kode 6.7: Beregning av D-delen for motor A, $D_A(k)$.

```
19     e_TurtallA_IIR(k) = IIR_filter(e_TurtallA_IIR(k-1), ...
                                         e_TurtallA(k), alfa2);
20     D_A(k) = Derivation(Kd*e_TurtallA_IIR(k-1:k), Ts(k-1));
```

Til dette bruker vi funksjonen `Derivation` fra kapittel 3, med en α -verdi `alfa2`.

- Totalpådraget $PowerA(k)$ beregnes til slutt som summen av alle delpådragene som vist i koden under

Kode 6.8: Kode for totalpådraget $PowerA(k)$ for motor A.

```
21     PowerA(k) = P_A(k)+I_A(k) + D_A(k);
```

I uttestingen av turtallsregulatoren, plottet vi alle bidragene $P_A(k)$, $I_A(k)$ og $D_A(k)$ i figurer slik at vi fikk innsikt i hvordan regulatoren fungerte.

Koden i dette kapittelet er plassert i `POX_F5_CalculateAndSetMotorPower.m`.

6.3 Resultat

6.3 Resultat

I dette kapittelet blir resultatene fra testing av turtallsregulatoren presentert. Først presenteres resultatene for P-regulatoren med dens begrensinger. Deretter presenteres resultatene for PI-regulatoren og vi dokumenterer behovet for integratorbegrensning. Til slutt presenteres resultatene for PID-regulatoren.

Resultatene består av tidsresponskurver som viser hvordan de forskjellige regulatorene klarer å opprettholde turtallet når LEGO-konstruksjonen blir løftet opp og ned fra bordet (på tilsvarende måte som vist i figur 6.6). For å stille inn regulatorene brukte vi prøve-og-feile metoden.

6.3.1 P-regulator

Den enkleste regulatoren er en P-regulator, og den består kun av koden vist i kodeutdrag 6.5. De beste resultatene oppnådde vi ved å bruke $\alpha=0.3$ og $K_p=1$, og et eksempel på en kjøring er vist i figur 6.11. Som vi ser fra delfigur b), klarer ikke P-regulatoren å følge referansen siden lengden på dobbelpilene markert med røde ringer indikerer hvor stort reguleringsavviket $e_{TurtallA}$ er. I tidspunktet rundt den første røde ringen ($t \approx 12$ sekund) ligger motoren ned mot bordet og utsettes for friksjon som vist i delfigur d). Reguleringsavviket avlest i b) ved $t \approx 12$ sekund er ca

$$e_{TurtallA} \approx 65\% - 20\% = 45\% \quad (6.4)$$

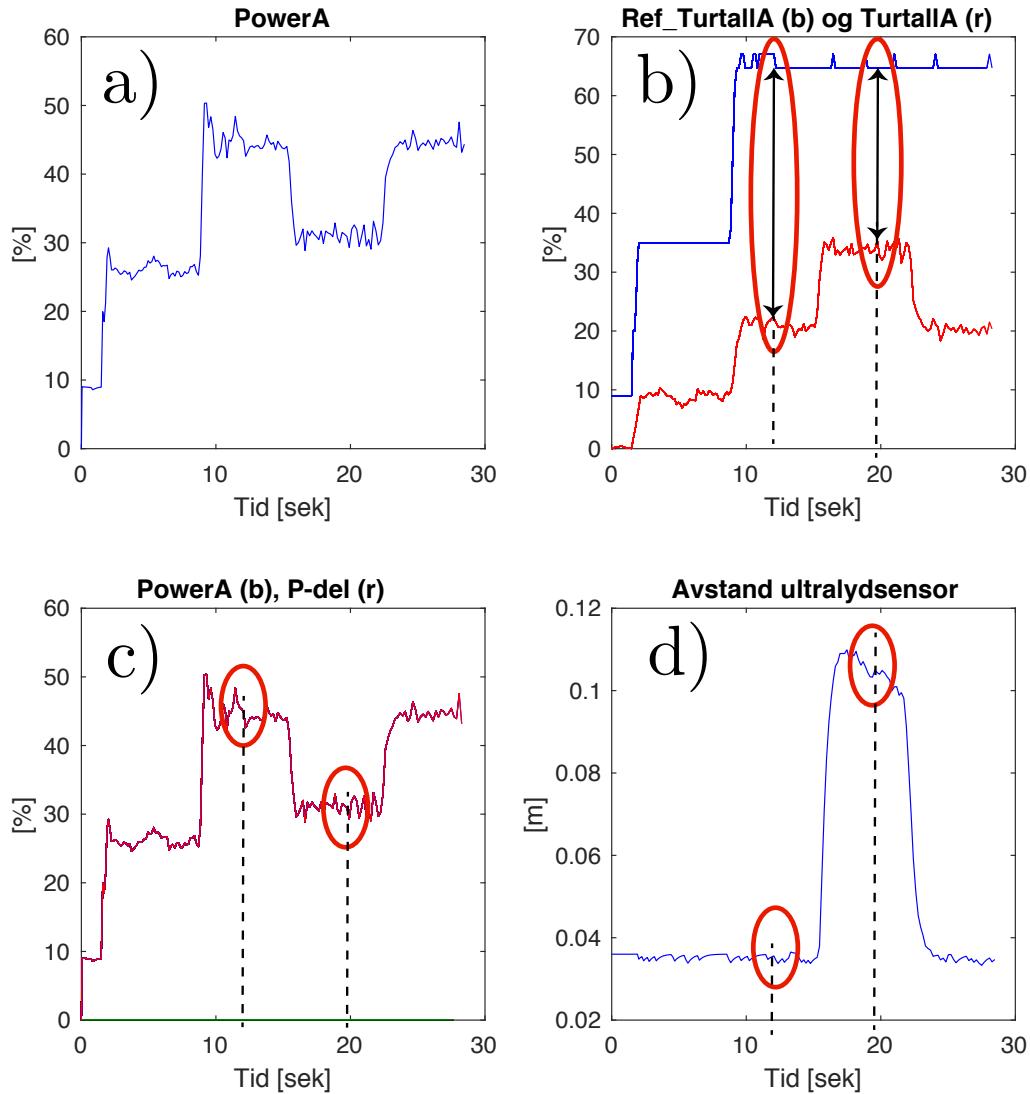
Den avleste pådragsverdien fra P-delen i c) ved samme tidspunkt er ca 45% (første røde ringen). Dette stemmer med uttrykket for en P-regulator når $K_p=1$:

$$P_A = K_p \cdot e_{TurtallA} \quad (6.5)$$

$$= 45 \quad (6.6)$$

En viktig observasjon fra ligning (6.5) som er gyldig for P-regulatorer i sin alminnelighet er at for å oppnå et pådrag som i det hele tatt er større enn 0, må reguleringsavviket også være større enn 0. Eller sagt med ord: Det må være avvik mellom ønskelig og virkelig turtall for at motoren i det hele tatt skal rotere.

6.3 Resultat



Figur 6.11: Målinger og beregninger fra et eksperiment som viser at P-regulatoren ikke klarer å følge referansen, verken med eller uten friksjon. Følgende parametere benyttet: $\alpha_1=0.3$, $K_p=1$, $K_i=0$, $K_d=0$. a): Beregnede pådragsverdier fra P-regulatoren. b): Referanse fra potensiometerert (blått) og beregnet turtall [%] (rødt). c): Bidragene fra P-del (rødt) og totalpådraget PowerA (blått). d): Avstandsmålinger fra ultralydsensoren. Forklaring til resultatene er gitt i teksten.

6.3 Resultat

Når LEGO-konstruksjonen løftes opp fra bordet ved $t \approx 20$ sekund som vist i d), reduseres reguleringsavviket til $e_{TurtallA} \approx 30$. Årsaken til dette er at når friksjonen forsvinner så roterer motoren lettere og turtallet øker (rød kurve i b)). Konsekvensen er altså at reguleringsavviket reduseres (kortere dobbelpil i b)), men dermed reduseres også pådraget (rød kurve i c)). Denne oppførselen stemmer med ligning (6.5).

For å oppsummere, så viser resultatene at en P-regulator ikke er i stand til å holde motorturtallet på en gitt referanseverdi, verken når den er påvirket eller upåvirket av friksjon. Ved å øke verdien av K_p vil reguleringsavviket reduseres, men det vil aldri bli lik 0 som vi egentlig ønsker. For å få fjernet reguleringsavviket må vi benytte en PI-regulator, se neste delkapittel.

6.3.2 PI-regulator

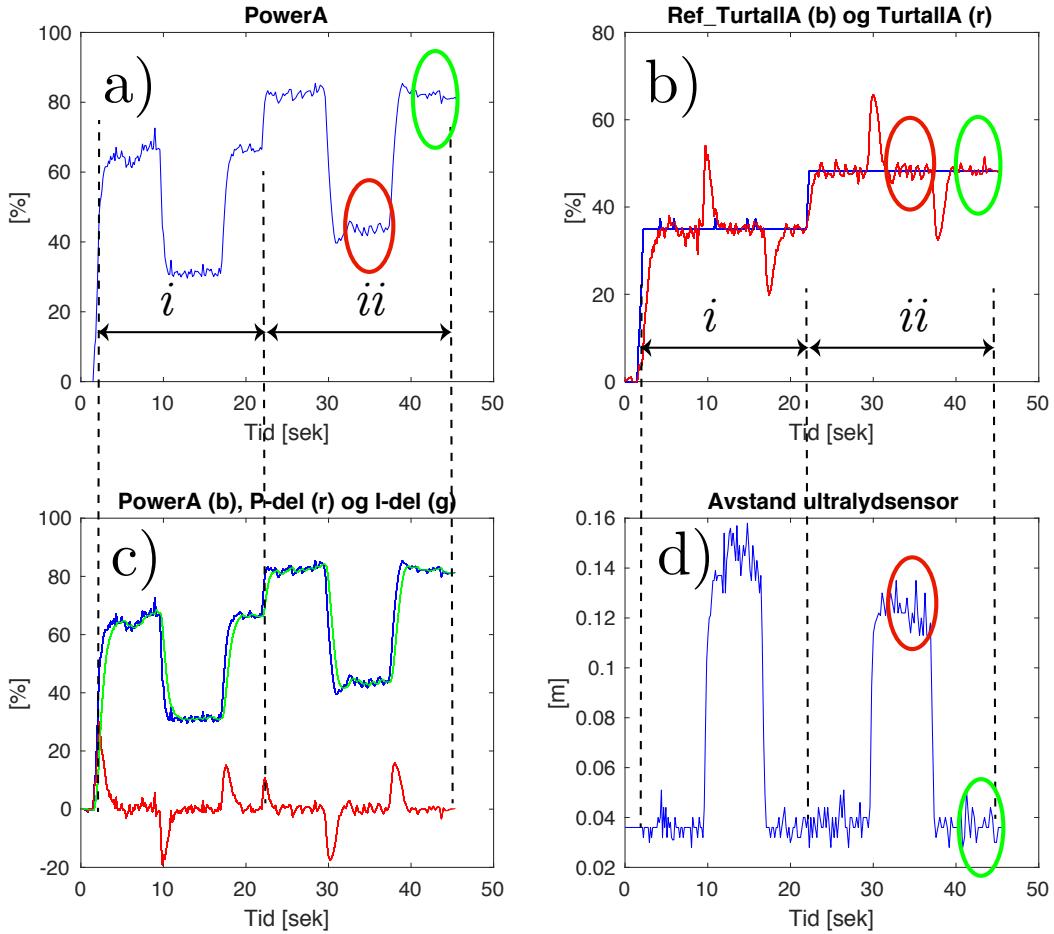
Ved å gjennomføre et lignende eksperiment som for P-regulatoren, fikk vi resultatene vist i figur 6.12, hvor $\alpha_1 = 0.3$, $K_p = 1$ og $K_i = 2$. Resultatene er delt inn i tidsperiodene *i* og *ii* som markerer to forskjellige referanseverdier, se blå kurve i b).

For å beskrive funksjonen til en PI-regulator, henvises det til tidsperiode *ii*. Når LEGO-konstruksjonen er løftet opp fra bordet, indikert med rød ring i d), er totalpådraget i a) ca 45% samtidig som både turtallet og referanseverdien i b) på ca 50%. Dette betyr at regulatoren sørger for at virkelig turtall blir lik ønsket turtall, og at reguleringsavviket

$$e_{TurtallA} \approx 0 \quad (6.7)$$

Samtidig ser vi at P-delen markert med rød kurve i c) ikke bidrar til turtallet (er tilnærmet lik 0), noe som er forventet ut fra ligning (6.5). Videre ser vi at det som bidrar mest til pådraget PowerA (som i a) og c) er vist med blå kurve) er I-delen markert som grønn kurve i c).

6.3 Resultat



Figur 6.12: Målinger og beregninger fra et eksperiment som viser hvordan turtallregulatoren kompenserer for friksjon. Følgende parametere ble benyttet: $\alpha_1=0.3$, $K_p=1$, $K_i=2$, $K_d=0$. a): Beregnede pådragsverdier fra PI-regulatoren. b): Referanse fra potensiometerert (blått) og beregnet turtall [%] (rødt). c): Bidragene fra P-del (rødt) og I-del (grønt), og totalpådraget PowerA (blått). d): Avstandsmålinger fra ultralydsensoren. Forklaring til resultatene er gitt i teksten.

Når LEGO-konstruksjonen legges ned på bordet, markert med grønn ring i d), ser vi fra a) at regulatoren øker pådraget, men at turtallet etter en kort innsvingningsperiode følger referansen, markert med grønn ring i b).

6.3 Resultat

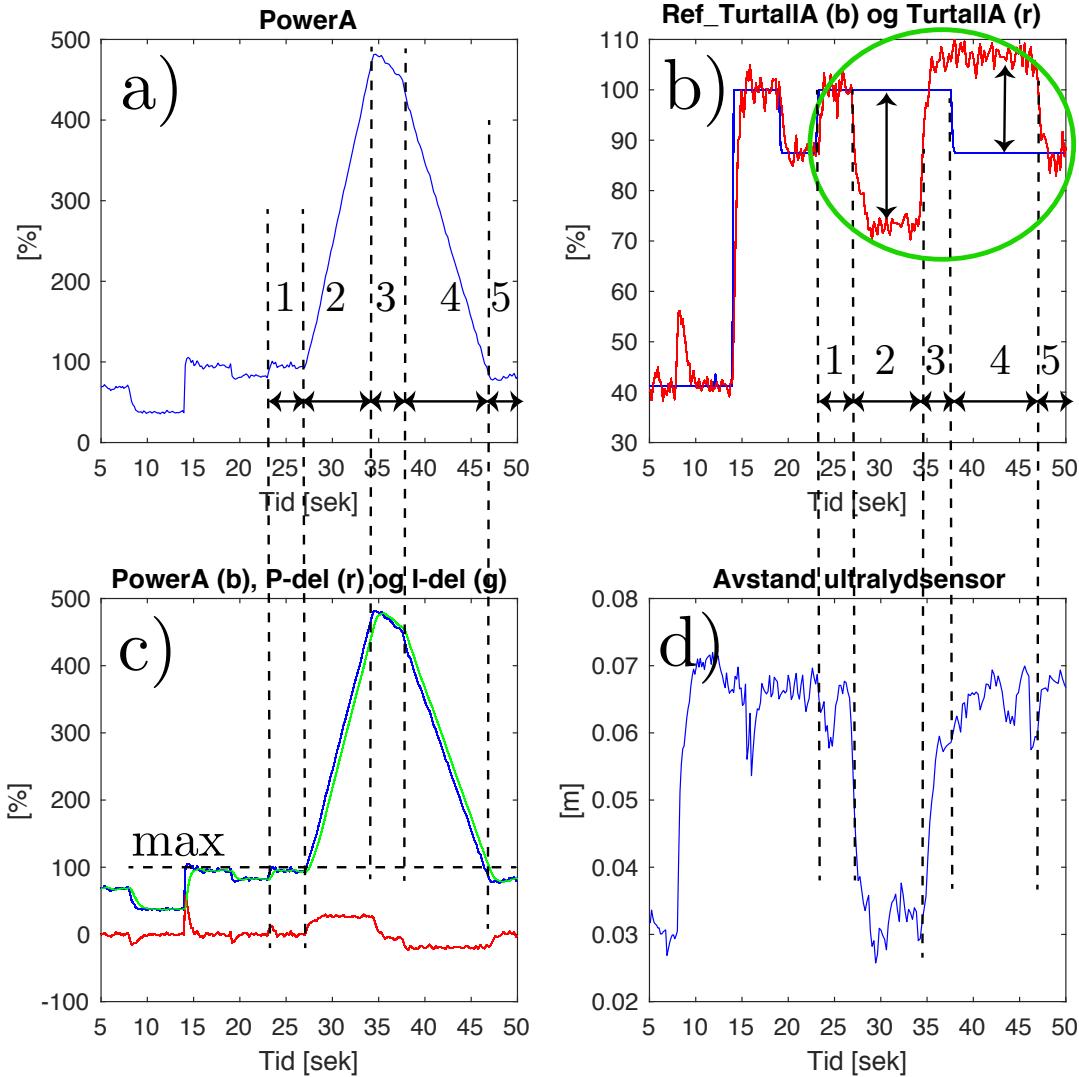
6.3.3 Integratorbegrensning

I dette delkapittel skal vi vise hva som skjer med en standard PI-regulator når regula-toren ikke klarer å holde utgangen lik referansen. En slik situasjon kan vi oppnå ved å sette referansen til 100% for deretter å legge LEGO-konstruksjonen ned på bordet. Friksjonen vil da bidra til at LEGO-motoren ikke klarer å holde 100% turtall, og dette er vist i grønn sirkel i figur 6.13b), hvor tidsperioden $23 < t < 50$ sekund er delt opp i følgende 5 delperioder hvor informasjonen lest ut fra delfigurene er listet opp i en slags kronologisk rekkefølge:

1. Gjelder for tidsperioden $23 < t < 27$.
 - d) LEGO-konstruksjonen er løftet opp og motoren er uten friksjon
 - b) Referansen og utgang er begge 100% turtall og reguleringsavviket er dermed 0
 - c) Pådraget er rett i underkant av 100% (markert med max).
2. Gjelder for tidsperioden $27 < t < 35$.
 - d) LEGO-konstruksjonen hviler på bordet og motoren utsettes for friksjon.
 - b) Referansen er fortsatt 100% mens virkelig turtall faller til ca. 75% turtall. Reguleringsavviket er dermed på 25% (vist med loddrett dobbelpil)
 - c) Dette resulterer i at I-leddet integreres lineært.
 - a) Beregnet teoretisk pådrag øker tilsvarende opp til ca 500%, men pådraget er i realiteten "kun" 100%.
3. Gjelder for tidsperioden $35 < t < 38$.
 - d) LEGO-konstruksjonen er løftet opp og motoren er uten friksjon.
 - b) Referansen er fortsatt 100% mens virkelig turtall stiger til ca. 105% turtall som er det turtallet motoren kan oppnå ubelastet ved fullt pådrag (tilsvarer vinkelhastighet på 857 avlest i delfigur c) i figur 6.8). Siden virkelig turtall er noe større enn referansen, er reguleringsavviket negativt, altså
$$e_{\text{TurtallA}} \approx -5\% \quad (6.8)$$
 - c) I denne perioden synker I-leddet fra ca 480 til ca 430
 - a) Totalpådraget synker tilsvarende som I-leddet

Når figurene inneholder mye data som er vanskelig for leseren å intuittivt forstå på egen hånd, er det lurt å gi en god og detaljert forklaring på hva resultatene viser. Dette viser også at DU har forstått hva som skjer. Dersom du ikke gir gode forklaringer på resultater som er vanskelige å forstå, vil ikke leseren heller forstå det og det teller negativt inn på karakteren.

6.3 Resultat



Figur 6.13: Målinger og beregninger fra et eksperiment som viser hva som skjer der som integratorene ikke begrenses når turtallregulatoren ikke klarer å følge referansen. Følgende parametere ble benyttet: $\alpha_1=0.3$, $K_p=1$, $K_i=2$, $K_d=0$. a): Beregnede pådragsverdier fra PI-regulatoren. b): Referanse fra potensiometerert (blått) og beregnet turtall (rødt). c): Bidragene fra P-del (rødt) og I-del (grønt), og totalpådraget PowerA (blått). d): Avstandsmålinger fra ultralydsensoren. Forklaring til resultatene er gitt i teksten.

6.3 Resultat

4. Gjelder for tidsperioden $38 < t < 47$.

- d) LEGO-konstruksjonen er fortsatt løftet opp og motoren er uten friksjon.
- b) Referansen er redusert til ca. 90% mens virkelig turtall fortsatt er 105% turtall
Dette gir et større negativt reguleringsavvik på

$$e_{\text{Turtalla}} \approx -15\% \quad (6.9)$$

- c) Dette gjør at I-leddet reduseres raskere enn i periode 3. Siden I-leddet fortsatt er større enn max pådrag på 100% ligger turtallet over referansen i hele denne perioden.⁵ Legg merke til å P-delen er negativ og den prøver å redusere turtallet.

5. Gjelder for tidsperioden $47 < t < 50$.

- d) LEGO-konstruksjonen er fortsatt løftet opp og motoren er uten friksjon
- b) Referansen er fortsatt ca. 90%, men nå er virkelig turtall også redusert til 90% og reguleringsavviket er 0.
- c) I-leddet er nå redusert til under 100%. P-leddet er ca 0% siden reguleringsavviket er 0.

Som eksperimentet viser, oppstår problemet i periode 2 hvor I-leddet integreres etter at pådraget er gått i metning og motoren ikke lenger klarer å følge referansen. Siden integratoren "må tømmes" med negativt reguleringsavvik, vil virkelig turtall ligge over ønsket turtall i periode 3 og 4, og dette er en uønsket egenskap. Løsningen er å forhindre at I-leddet integreres opp etter at I-leddet har passert 100% nivå, og dette løste vi som vist i kodeutdrag 6.9.

Kode 6.9: Kode for integratorbegrensning i turtallsregulatoren.

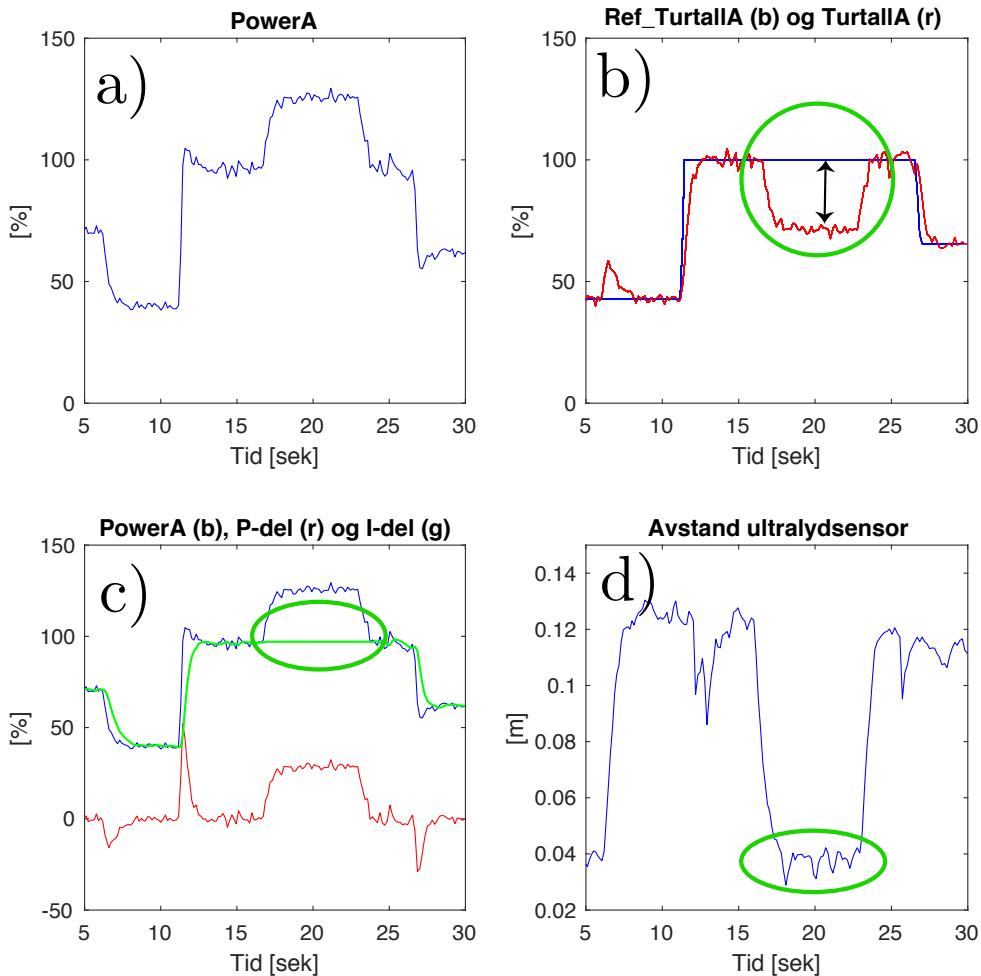
```
18     I_A(k) = EulerForward(I_A(k-1), Ki*e_Turtalla(k-1), Ts(k-1));
19     if abs(I_A(k))>100
20         I_A(k)=I_A(k-1);
21     end
```

Ved å benytte denne koden i et lignende eksperiment som vist i figur 6.13, fikk vi resultatene vist i figur 6.14 hvor de grønne ringene i markerer effekten av integratorbegreningen. I delfigur d) blir LEGO-konstruksjonen blir lagt ned på bordet i tidsperioden $17 < t < 24$. Samtidig faller turtallet i b) på samme måte som i periode 2

⁵Hadde dette vært en *cruise controller* til en svært tung bil som i en oppoverbakke ikke klarte å holde fartsgrensen, ville bilen kjørt mye over fartsgrensen når den passerte toppen av bakken.

6.3 Resultat

i figur 6.13 hvor på det blir et vedvarende regulatingsavvik større enn 0, men integratorbegreningen gjør at I-leddet stopper på 100% i c). Når LEGO-konstruksjonen løftes opp fra bordet ved $t > 24$, går turtallet opp til referansen med en gang, og regulatingsavviket er 0.

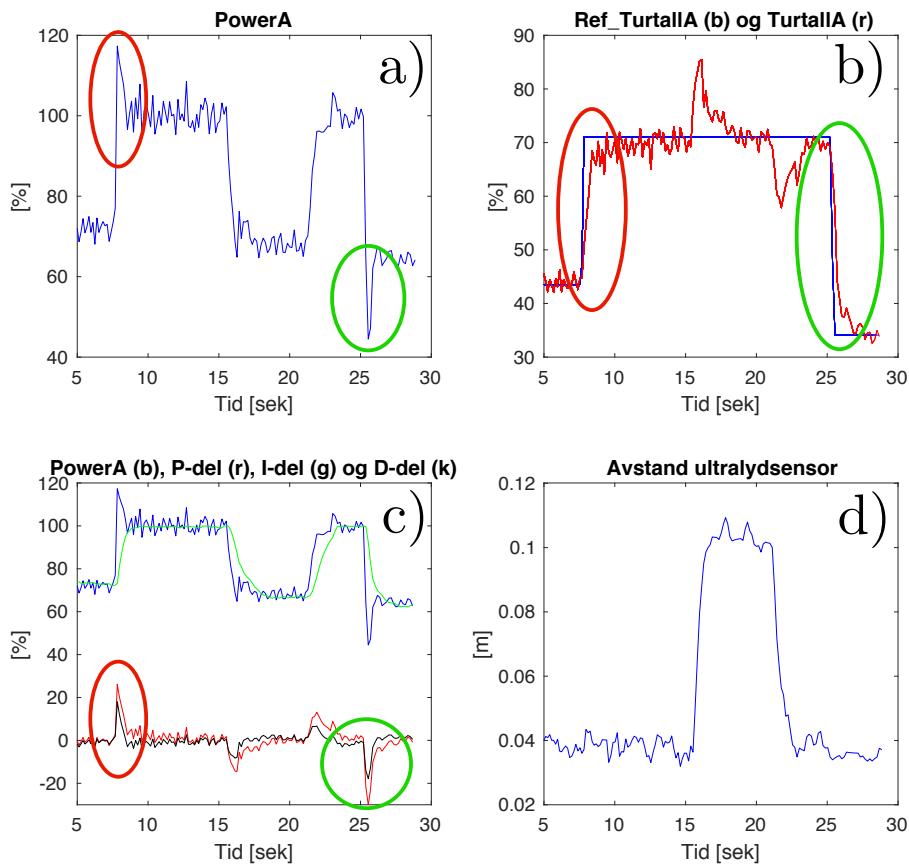


Figur 6.14: Målinger og beregninger fra et eksperiment som viser effekten av integratorbegrensning. Følgende parametre ble benyttet: $\alpha_1=0.3$, $K_p=1$, $K_i=2$, $K_d=0$. a): Beregnede pådragsverdier fra PI-regulatoren. b): Referanse fra potensiometerert (blått) og beregnet turtall (rødt). c): Bidragene fra P-del (rødt) og I-del (grønt), og totalpådraget PowerA (blått). d): Avstandsmålinger fra ultralydsensoren. Forklaring til resultatene er gitt i teksten.

6.3 Resultat

6.3.4 PID-regulator

Ved å legge til D-delen av PID-regulatoren, vil totalpådraget få et bidrag som er bestemt ut fra hvor fort reguleringsavviket endres. Resultatet fra et forsøk med en PID-regulator er vist i figur 6.15, hvor den største effekten av D-delen er markert med grønne og røde ringer.



Figur 6.15: Målinger og beregninger fra et eksperiment som viser hvordan PID-versjonen av turtallregulatoren kompenserer for friksjon. Følgende parametre ble benyttet: $\alpha_1=0.3$, $K_p=1$, $K_i=2$, $K_d=1$, $\alpha_2=0.1$. a): Beregnede pådragsverdier fra PID-regulatoren. b): Referanse fra potensiometerert (blått) og beregnet turtall (rødt). c): Bidragene fra P-del (rødt), I-del (grønt), D-del (svart) og totalpådraget PowerA (blått). d): Avstandsmålinger fra ultralydsensoren. Forklaring til resultatene er gitt i teksten.

6.3 Resultat

Følgende regulatorparametre er benyttet; $K_p=1$, $K_i=2$, $K_d=1$. I filteret som benyttes i derivatdelen benyttet vi $\alpha_2=0.1$.

Siden bidraget fra D-delen gjør seg gjeldene når reguleringsavviket endres, ser vi at den positive referanseendringen ved $t = 7$ sekund (rød ring, delfigur b)) bidrar til at D-delen gir en positivt, men kortvarig bidrag (delfigur c)). Dette gjør at turtallet stiger raskere enn det ville gjort uten D-delen. Legg merke til at i a) så går totalpådraget over 100% som betyr at regulatoren egentlig ikke får fullt utnyttet effekten av D-delen.

Derimot så viser responsen markert med grønne ringer et eksempel på at regulatoren får fullt utbytte av D-delen. Den negative referanseendringen gjør at D-delen bidrar med negative verdier (vist i c)), og at totalpådraget i a) går helt ned mot 40% pådrag før den stabiliserer seg på 65% mot slutten. Effekten på turtallet er at det faller raskere enn uten D-del.

En siste viktig, men negativ, bi-effekt av D-delen er at pådraget blir mer støyfylt når reguleringsavviket deriveres. Dette kan reduseres ved å filtrere enda kraftigere (lavere verdi på α_2).

6.3.5 Effekten av å kople inn turtallsregulatoren

For å få vist effekten av å kople inn og ut turtallsregulatoren under kjøring, bestemte vi oss for å benytte potensiometeret både som direkte pådrag (i "manuell") og som referanse (i "auto"). Dette er vist i kodeutdrag 6.10 hvor vi bestemmer funksjonen til turtallsregulatoren ved å trykke på en LEGO-trykkbryter. Er den inntrykket er regulatoren i manuell og pådraget til motoren settes til potensiometerverdien. Dette er vist i linje 16 i koden under.

For å unngå problemer i plottingen av dataene ved at det mangler verdier i vektorene, setter vi $P_A(k)=NaN$ og $D_A(k)=NaN$. I tillegg justerer vi kontinuerlig integratorleddets verdi som vist i linje 18. Dermed unngår vi at I-leddet integrerer seg opp eller ned mens vi kjører i manuell, og i det vi kopler over til automatisk kjøring, vil regulatoren starte med samme pådrag som den hadde i manuell. Dette gir en såkalt støtfri omkoppling, eller *bumpless transfer*.

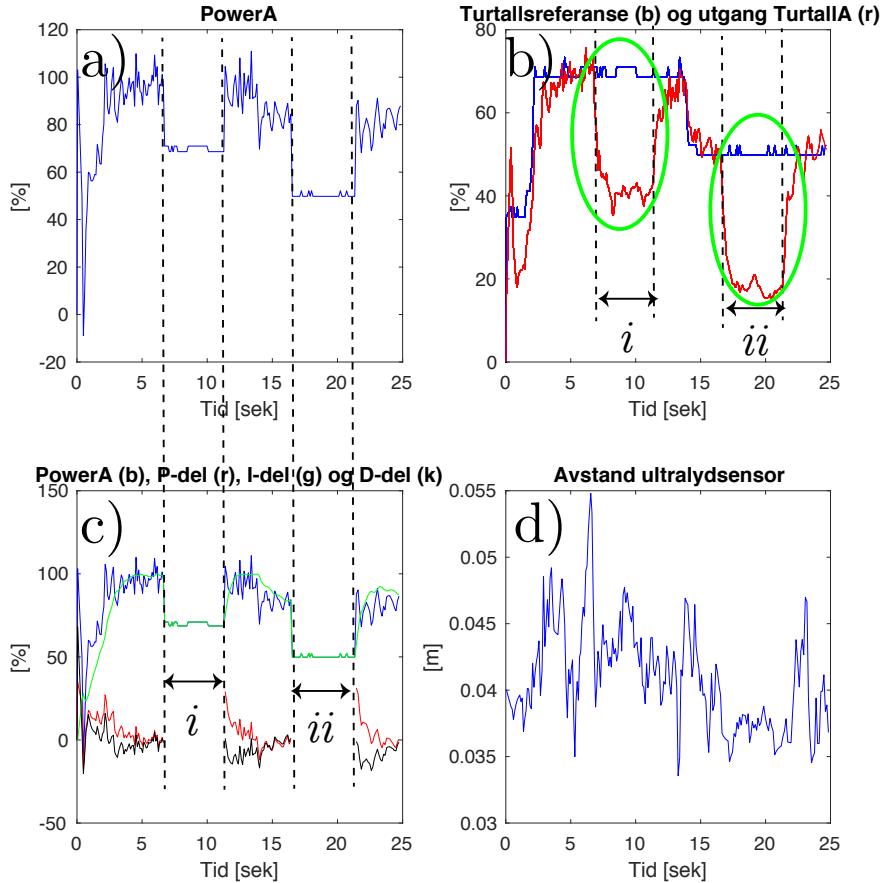
6.3 Resultat

Kode 6.10: Kode for turtallsregulator for motor A.

```
11 % trykknapp bestemmer om regulator er i auto eller manuel
12 TurtallsReg(k) = ~Bryter(k);
13 if TurtallsReg(k)
14     PowerA(k) = P_A(k)+I_A(k) + D_A(k);
15 else
16     PowerA(k) = PotMeter(k);
17     P_A(k) = NaN;
18     I_A(k) = PowerA(k);
19     D_A(k) = NaN;
20 end
21 motorA.Speed = PowerA(k);
22 start(motorA);
```

Resultatet av eksperimentet er vist i figur 6.16, hvor LEGO-konstruksjonen har hvilt på bordet under hele eksperimentet (delfigur d)). Regulatoren er koplet ut i tidsperiodene merket med *i* og *ii*. Vi observerer at turtallet faller i disse tidsperiodene, markert med grønne ringer i delfigur b).

6.3 Resultat



Figur 6.16: Målinger og beregninger fra et eksperiment som viser effekten av å koppe inn og ut turtallsregulatoren. Turtallsregulatoren er utkoplet i tidsperiodene merket med *i* og *ii*. Følgende parametere ble benyttet: $\alpha_1=0.3$, $K_p=1$, $K_i=2$, $K_d=1$, $\alpha_2=0.1$. a): Beregnede pådragsverdier fra PID-regulatoren, samt manuelt pådrag i periodene *i* og *ii*. b): Referanse fra potensiometerert (blått) og beregnet turtall (rødt). c): Bidragene fra P-del (rødt), I-del (grønt), D-del (svart) og totalpådraget PowerA (blått). Legg merke til at P-del og I-del er NaN i perioden med manuelt pådrag. I-delen beholdes som manuelt pådrag for å unngå store endringer ved innkoppling. d): Avstandsmålinger fra ultralydsensoren viser at LEGO-konstruksjonen har ligget på bordet under hele forsøket.

6.4 Turtallsregulator som funksjon

6.4 Turtallsregulator som funksjon

For å kunne bruke turtallsregulatorkoden på en enkel måte i prosjektene våre, laget vi en funksjon av koden. Vi har brukt *persistent-funksjonaliteten* i MATLAB, og kan på den måten spare endel oversending av data i funksjonskallet.

Bruk av *persistent* vil bli gått gjennom i undervisningen.

Siden variable som blir definert som *persistent* vil ligge tilgjengelig lokalt i funksjonen til neste gang den blir kallet, vil det ikke fungere å bruke samme funksjon til å beregne pådrag for flere motorer. Dette fordi tallverdiene for de forskjellige motorene vil bli blandet. Av den grunn har vi laget tre identiske funksjoner kalt `CalcPowerA.m`, `CalcPowerB.m` og `CalcPowerC.m`.

Under blir koden til `CalcPowerA.m` vist. I selve funksjonskallet sendes

- motorobjektet, for avlesing av vinkelposisjon inne i selve funksjonen
- referansen, som tilsvarer enten ønsket turtall eller manuelt pådrag
- regulatorfunksjonalitet, som P, PI, PD, I, PID eller manuell.
- regulatorparametre

Deretter spesifiseres de variable som er lokale for funksjonen, og standardverdier for regulatorparametre, se kodeutdrag 6.11.

Kode 6.11: Kode fra funksjonen for turtallsregulering av motor A.

```
18 function [u,varargout] = ...
    CalcPowerA(motorA, SpeedRef, action, varargin)
19
20 persistent e e_IIR angle angle_IIR timestamp I
21
22 % default values of parameters
23 Kp      = 1;
24 Ki      = 2;
25 Kd      = 1;
26 umax   = 100;
27 umin   = -100;
28 alfa1  = 0.3; % angular velocity filtering
29 alfa2  = 0.1; % error filtering in D-part
```

6.4 Turtallsregulator som funksjon

Dersom brukeren ønsker å sende inn andre regulatorparametere, kan dette sendes inn som `varargin` som inneholder en struct med parameterverdier. For å overskrive standardverdiene vist over, brukes følgende kode:

Kode 6.12: Kode fra funksjonen for turtallsregulering av motor A.

```
31 % If you have specified some parameters in main file, they ...
      will overwrite
32 % the default values in this function
33 if nargin>3
34     paranew=varargin{1};
35     if isfield(paranew,'Kp')
36         Kp = paranew.Kp;
37     elseif isfield(paranew,'Ki')
38         Ki = paranew.Ki;
39     elseif isfield(paranew,'Kd')
40         Kd = paranew.Kd;
41     elseif isfield(paranew,'umax')
42         umax = paranew.umax;
43     elseif isfield(paranew,'umin')
44         umin = paranew.umin;
45     elseif isfield(paranew,'alfal')
46         alfaf1 = paranew.alfal;
47     elseif isfield(paranew,'alfa2')
48         alfa2 = paranew.alfa2;
49     end
50 end
```

Ved bruk av *persistent* variable, trenger vi ikke ta vare på mer enn de 2 siste elementene i variablene. Ved å spesifisere element nr 2 i disse variablene til 0 som vist i kodeutdraget under, blir også element nr 1 satt lik 0 dersom den ikke allerede har et innhold.

Ved aller første gangs kall av funksjonen har ikke element nr 1 innhold, men ved å flytte element nr 2 til element nr 1 i slutten av funksjonen (samme som å gå ett tidsskritt frem), vil element nr 1 ha innhold ved neste kall, og element nr 1 blir derfor ikke overskrevet til 0.

Kode 6.13: Kode fra funksjonen for turtallsregulering av motor A.

```
31 angle(2)=0;
32 angle_IIR(2)=0;
33 e(2)=0;
34 e_IIR(2)=0;
35 I(2)=0;
```

6.4 Turtallsregulator som funksjon

Resten av koden tilsvarer det som er vist i kodeutdragene 6.2, 6.3, 6.4, 6.5, 6.6, 6.7 og 6.8, og er vist i sin helhet under.

Kode 6.14: Kode fra funksjonen for turtallsregulering av motor A.

```
31 timestamp(2) = toc;
32 Ts = timestamp(2)-timestamp(1);
33 angle(2)      = double(motorA.readRotation);
34 angle_IIR(2)  = IIR_filter(angle_IIR(1),angle(2),alfa1);
35 angular_speed = Derivation(angle_IIR, Ts);
36 speed = 1/8*angular_speed;
37 e(2) = SpeedRef - speed;
38 P = Kp*e(2);
39 I(2) = EulerForward(I(1), Ki*e(1), Ts);
40 if I(2)>umax
41     I(2)=I(1);
42 elseif I(2)<umin
43     I(2)=I(1);
44 end
45 e_IIR(2) = IIR_filter(e_IIR(1), e(1), alfa2);
46 D = Derivation(Kd*e_IIR, Ts);
```

Ut fra regulatorfunksjon, bestemmes deretter hvilket pådrag som skal beregnes. Valgene er P-regulator, PI, PID, PD, I eller manuell, se under.

Kode 6.15: Kode fra funksjonen for turtallsregulering av motor A.

```
31 if strcmp(action,'P')
32     u = P;
33     I(2)=NaN;
34     D=NaN;
35 elseif strcmp(action,'PI')
36     u = P + I(2);
37     D=NaN;
38 elseif strcmp(action,'PID')
39     u = P + I(2) + D;
40 elseif strcmp(action,'PD')
41     u = P + D;
42     I(2)=NaN;
43 elseif strcmp(action,'I')
44     u = I(2);
45     P=NaN;
46     D=NaN;
47 elseif strcmp(action,'man')
48     u = SpeedRef;
49     I(2)=SpeedRef;
50     P=NaN;
```

Siden innholdet i dette delkapittelet blir veldig mye kode og lite beskrivende tekst, merker du kanskje selv at du ikke bruker tid på å lese denne delen. Det beste ville vært å ha en liten introduksjon til tankegangen bak funksjonen, og legge hele koden i et vedlegg. Dersom du opplever at deler av din egen rapport ligner på akkurat denne siden her, vil leseren mest sannsynlig bare hoppe over dette (sensor også). Jeg har gjort dette med vilje her bare for at du skal se hvordan rapporten IKKE skal se ut.

6.4 Turtallsregulator som funksjon

```
51      D=NaN;  
52  end
```

Til slutt flyttes dataene i element 2 til element 1, som innebærer det samme som å gå et tidsskritt frem. Samtidig gjøres variablene i **varargout** klar, se under.

Kode 6.16: Kode fra funksjonen for turtallsregulering av motor A.

```
31 angle(1)      = angle(2);  
32 angle_IIR(1) = angle_IIR(2);  
33 e(1)          = e(2);  
34 e_IIR(1)     = e_IIR(2);  
35 timestamp(1) = timestamp(2);  
36 I(1)          = I(2);  
37  
38 outputarg = [P,I(2),D];  
39 nout = max(nargout,1) - 1;  
40 for k = 1:nout  
41     varargout{k} = outputarg(k);  
42 end
```

Kapittel 7

Litt om referering til rapportelementer

7.1 Tagging av rapportelement, `\label{ }`

For at du i teksten du skriver skal kunne referere til forskjellige rapportelementer må du legge til en *tag* eller *bookmark* som tydelige identifiserer selve elementet. Til dette bruker du kommandoen `\label{ }`, hvor det som står inne i krøllparentesene er den unike referanseteksten/identifikasjonen for elementet. Siden det finnes veldig mange forskjellige rapportelementer som det kan refereres til, er det vanlig å bygge opp selve referanseteksten slik:

```
\label{forkortelse for elementet:beskrivende tekst}
```

Eksempler på typiske rapportelementer og forkortelser er vist i listen under, hvor `xxxxx` er satt inn for beskrivende tekst.

1. For å referere til **kapitler** benyttes ofte `kap` som forkortelse¹, og `\label{kap:xxxxx}` må plasserers innenfor selve kapittelet, og typisk rett bak `\chapter{Kapitteloverskrift}`. Helt øverst i selve .tex-fila til dette kapittelet, `litt_om_referering.tex`, vil du se måten det er gjort på her:
`\chapter{Litt om referering til rapportelementer}\label{kap:referering}`

¹Dersom du vil bruke engelske ord er `ch` for `\chapter` et godt alternativ.

7.1 Tagging av rapportelement, `\label{ }`

2. For å referere til **delkapitler** benyttes ofte `delkap` som forkortelse², og ellers gjøres det på samme måte som for kapitler. For dette delkapittelet (kapittel 7.1) er følgende *tag* brukt `\label{delkap:label}`.
3. For å referere til **ligninger** benyttes ofte `eq` som forkortelse, og `\label{eq:xxxxx}` plasseres mellom `\begin{equation}` og `\end{equation}` som vist under

```
\begin{equation}
    \label{eq:sinus} <-----
    y(t)=\sin(\omega t)
\end{equation}
```

Dette blir vist mer i detalj i kapittel 8.

4. For å referere til **figurer** benyttes ofte `fig` som forkortelse, og `\label{fig:xxxxx}` plasseres mellom `\begin{figure}` og `\end{figure}` som vist under

```
\begin{figure}[H]
    \centering
    \scalebox{0.6}{\includegraphics{sinuskurve}}
    \caption{Figuren viser en sinuskurve som funksjon av tid.}
    \label{fig:sinuskurve} <-----
\end{figure}
```

Dette blir vist mer i detalj i kapittel 9.

5. For å referere til **tabeller** benyttes ofte `tab` som forkortelse, og `\label{tab:xxxxx}` plasseres mellom `\begin{table}` og `\end{table}` som vist under

```
\begin{table}[H]
    \centering
    \caption{Enkel tabell.}
    \begin{tabular}{|c|c|}\hline
        a & b & \\ \hline\hline
        2 & 0.6 & \\ \hline
    \end{tabular}
    \label{tab:a_og_b} <-----
\end{table}
```

Dette blir vist mer i detalj i kapittel 10.

6. For å referere til **\item-elementer** i en punktliste benyttes ofte `pkt` som forkortelse, og `\label{pkt:xxxxx}` plasseres bak teksten som tilhører punktet. Du kan IKKE bruke dette i `\begin{itemize}` og `\end{itemize}` siden

²Dersom du vil bruke engelske ord er `sec` for `\section` et godt alternativ.

7.2 Tagging av kode, `label=`

punktene er svarte, unummererte prikker. Du kan kun referere i lister laget av `\begin{enumerate}` og `\end{enumerate}` som vist i eksempelet under

```
\begin{enumerate}
\item Biler er gule.
\item Roser er røde. \label{pkt:roses}
\end{enumerate}
```

Dette blir vist mer i detalj i kapittel 11.

7. For å referere til **sidenummer** benyttes ofte `side` som forkortelse, og `\label{side:xxxxx}` plasserer rett i teksten og nær teksten du ønsker å referere til.

Legg merke til at den beskrivende teksten kan være hva som helst, og gjerne identisk med andre beskrivende tekster. Sammenkoplingen av forkortelser av elementer og beskrivende tekst gjør alikevel selve referansen helt unik. **Husk at du aldri må bruke æ,ø eller å i referansetekstene.**

7.2 Tagging av kode, `label=`

Dersom du bruker `listings`-pakken slik det blir beskrevet i kapittel 12, brukes `label` på en litt annen måte. Istedentfor å skrive `\label{kode:xxxxx}` skal du heller skrive `label=kode:xxxxx`. Denne skal plasseres i hakparentes sammen med `caption` og eventuelle linjenummerspesifikasjoner som vist i følgende eksempel:

```
\lstinputlisting[firstnumber=14,firstline=14,lastline=28,
                caption={Utdrag av funksjonen {\tt SaveMyFigure.m}.},
                label=kode:Utdrag_av_SaveMyFigure]{SaveMyFigure.m}
```

Dette eksempelet viser kodelinjene 14-28 fra filen `SaveMyFigure.m` som ligger i samme mappe som hovedfilen `Gruppe19XX.tex`. Resultatet av denne koden er vist i kodeutdrag 12.1 på side 68.

Dette blir vist mer i detalj i kapittel 12.

7.3 Bruk av `\ref{ }`, `\eqref{ }` og `\pageref{ }`

7.3 Bruk av `\ref{ }`, `\eqref{ }` og `\pageref{ }`

For å referere til kapitler, ligninger, figurer, tabeller og punktlister brukes `\ref{ }`.

For å referere til *ligninger* bør du bruke `\eqref{ }` som sørger for at parentesene som omslutter ligningsnummeret blir med i teksten. For å referere til *sidenummer* må du bruke `\pageref{ }`.

Under er vist et overdrevet eksempel på referering til en ligning gitt på en side i et kapittel (vanligvis holder det å referere kun til ligningsnummeret).

Som vist i ligning`\eqref{eq:IIR-filter}` på side`\pageref{side:integrasjon}`, gitt i kapittel`\ref{kap:integrasjon}`, benyttes et IIR-filter for å beregne filtrert lys.

Bruken av tilde mellom ordet `ligning` og koden `\eqref{eq:IIR-filter}` er for å knytte disse to elementene sammen slik at de ikke kan deles/separeres. På den måten unngår du at ny linje begynner med et ligningsnummer (som kan forveksles med en slags nummerert liste).

7.4 Beskrivelse av litteratur

7.4 Beskrivelse av litteratur

For å kunne referere til litteratur, må du først lage en .bib-fil som du f.eks. kan kalle `referanser.bib` | denne filen ligger data om all litteratur du bruker, og dataene organiseres i en spesiell struktur som typisk ser slik ut:

```
@TypeLitteratur{Tag,  
    author = {},  
    title = {},  
    year = {},  
    +  
    +  
}
```

hvor

- `@TypeLitteratur` kan være bl.a. `@Book`, `@TechReport`, `@Misc`, `@Article`, `@InProceedings`, `Article`, `@PhDThesis`, `@MastersThesis` og `@Unpublished`³. Disse blir beskrevet i mer detalj nedenfor, men det er `@Book`, `@TechReport` og `@Misc` som er mest relevant for deg i ING100, mens de andre typene er mer aktuelle å bruke i bacheloroppgaven din.
- `Tag` er en *tag* på samme måte som andre rapportelementer beskrives med en unik nøkkel. Det er mange måter å bygge en slik *tag* opp på, men en relativt vanlig variant er at ta de 3 første bokstavene i etternavnet til hver forfatter og til slutt årstallet. Dersom det er mange forfattere kan *tag*'en bli veldig lang, og da pleier jeg å ta med 3 bokstaver fra etternavnet til første forfatter og deretter skrive `_et_al.` som er latisk forkortelse for "med flere".
- `author`, `title`, `year` og andre elementer varierer med hvilken type publikasjon vi snakker om. Dette blir vist mer i detalj nedenfor

Et eksempel på en .bib-fil ligger i samme mappe som hovedfilen `Gruppe19XX.tex`. Den heter `referanser.bib` og inneholder ett eksemplar av typene `@Book`, `@TechReport` og `@Misc`, og disse blir beskrevet i den neste 3 delkapitlene.

³Det finnes mange flere varianter.

7.4 Beskrivelse av litteratur

7.4.1 `@Book`

`@Book` bruker du når du skal referere til noe i en bok, f.eks. læreboka i MAT100 [2]. I `referanser.bib` ser den slik ut:

```
@Book{AdaEss2017,  
  author = {R. A. Adams and C. Essex},  
  title = {Calculus. A complete course},  
  publisher = {Pearson},  
  year = {2017},  
  edition = {9},  
}
```

Legg merke til at hver forfatter skiller med ordet *and*.

Den fullstendige beskrivelsen av `@Book` er

```
@Book{,  
  ALTAuthor = {},  
  ALTEditor = {},  
  title = {},  
  publisher = {},  
  year = {},  
  OPTkey = {},  
  OPTvolume = {},  
  OPTnumber = {},  
  OPTseries = {},  
  OPTaddress = {},  
  OPTedition = {},  
  OPTmonth = {},  
  OPTnote = {},  
  OPTannote = {}  
}
```

hvor elementene `ALTAuthor` og `ALTEditor` betyr at du må velge ett av alternativerne og samtidig fjerne ordet `ALT`. Ordet `OPT` som står foran de nederste elementene betyr *optional* og trenger ikke fylles ut. De eneste feltene som er obligatoriske er `author/editor`, `title`, `publisher` og `year`.

7.4 Beskrivelse av litteratur

7.4.2 @TechReport

@TechReport brukes for tekniske rapporter som verken er lærebøker eller artikler. Projektbeskrivelsen i ING100 kan kalles en teknisk rapport, og den referansen ser slik ut i referanser.bib.

```
@TechReport{Dre2019,
    author = {T. Drengstig},
    title = {Lego Mindstorms og MATLAB; anvendt matematikk og fysikk i skjønn forening},
    institution = {Universitet i Stavanger},
    year = {2019},
    note = {Utdelt materiale},
}
```

Den fullstendige beskrivelsen av @TechReport er

```
@TechReport{,
    author = {},
    title = {},
    institution = {},
    year = {},
    OPTkey = {},
    OPTtype = {},
    OPTnumber = {},
    OPTaddress = {},
    OPTmonth = {},
    OPTnote = {},
    OPTannote = {}
}
```

Ordet OPT som står foran de nederste elementene betyr *optional* og trenger ikke fylles ut. De eneste feltene som er obligatoriske er author, title, institution og year.

7.4 Beskrivelse av litteratur

7.4.3 @Misc

@Misc brukes for diverse ting, derfor typen *miscellaneous*. Den fullstendige beskrivelse er gitt under, hvor alle elementene er *optional*.

```
@Misc{,  
    OPTkey = {},  
    OPTauthor = {},  
    OPTtitle = {},  
    OPThowpublished = {},  
    OPTmonth = {},  
    OPTyear = {},  
    OPTnote = {},  
    OPTannote = {}  
}
```

Denne kan være nyttig til å referere til nettsteder slik som vist på dette nettstedet [1], hvor selve referansen ser slik ut i `referanser.bib`:

```
@Misc{web_BibTex,  
    title = {How can {I} use Bibtex to cite a web page?},  
    howpublished = {https://tex.stackexchange.com/questions/3587/how-can-i-use-bibtex-to},  
}
```

Legg merke til at for at "I" skal forbli en stor i tittelen når den gjengis i referanselista, må du bruke krøllparenteser rundt (som vist med {I}).

7.4 Beskrivelse av litteratur

7.4.4 @Article

@Article er en artikkel som er publisert i en journal . Et eksempel er denne <https://bit.ly/2SCNDdq> som er en såkalt oversiktsartikkel (*review-artikkel*) om bruken av LEGO i høyere utdanning de siste 15 år. Den er publisert i International Journal of Advanced Robotic Systems. Ved å trykke på **Download PDF**-knappen vil du se at artikkelen ser slik ut.

The screenshot shows a journal article page. At the top left is the journal title 'International Journal of Advanced Robotic Systems'. To the right is an 'OPEN ACCESS ARTICLE' logo. The main title of the article is 'LEGO-based Robotics in Higher Education: 15 Years of Student Creativity', described as an 'Invited Review Article'. Below the title, the authors listed are Ethan Danahy¹, Eric Wang², Jay Brockman³, Adam Carberry⁴, Ben Shapiro¹ and Chris B. Rogers^{1,*}. There are four small superscript numbers (1, 2, 3, 4) corresponding to author affiliations. A note indicates the * Corresponding author E-mail: Chris.Rogers@tufts.edu. Below the authors, the text 'Received 23 Aug 2013; Accepted 18 Jan 2014' and 'DOI: 10.5772/58249' is shown. A copyright notice states: '© 2014 The Author(s). Licensee InTech. This is an open access article distributed under the terms of the Creative Commons Attribution License (<http://creativecommons.org/licenses/by/3.0/>), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.' At the bottom, there are two columns of text: 'Abstract' and '1. Introduction'. The abstract begins with 'Our goal in this article is to reflect on the role LEGO robotics has played in college engineering education over the last 15 years, starting with the ...' and ends with 'LEGO robots have been used in hundreds of college-level ...'.

Figur 7.1: Artikkelen publisert i en journal.

7.4 Beskrivelse av litteratur

På grunn av antall forfattere, ville denne artikkelen fått *tag'en* `Dan_et_al.2014` i .bib-fila. Husk at du ikke benytter linjeskift i oppramsing av *authors*, se under

```
@Article{Dan_et_al.2014,
  author = {Ethan Danahy and Eric Wang and Jay Brockman and Adam Carberry and Ben Shapiro and Chris B.
            title = {LEGO-based Robotics in Higher Education: 15 Years of Student Creativity},
            journal = {Int J Adv Robot Syst},
            year = {2014},
            pages = {1--15},
            volume = {11},
            number = {27},
            note = {doi: 10.5772/58249}
}
```

Legg merke til at journalens navn er forkortet til Int J Adv Robot Syst, og at sidenummerområdet bruker 2 bindestreker.

Den fullstendige beskrivelsen av @Article er

```
@Article{,
  author = {},
  title = {},
  journal = {},
  year = {},
  OPTkey = {},
  OPTvolume = {},
  OPTnumber = {},
  OPTpages = {},
  OPTmonth = {},
  OPTnote = {},
  OPTannote = {}
}
```

7.4 Beskrivelse av litteratur

7.4.5 @InProceedings

@InProceedings er en artikkel som er presentert ved en konferanse. Et eksempel er denne <https://bit.ly/2AvqFO8> som handler om bruk av LEGO Mindstorms, og som ble presentert på IFAC sin 18. verdenskonferanse i Milano. Ved å trykke på **Download full text in PDF** vil du se at artikkelen ser slik ut.

**Proceedings of the 18th World Congress
The International Federation of Automatic Control
Milano (Italy) August 28 - September 2, 2011**



**A LEGO Mindstorms multi-robot setup in
the Automatic Control Telelab**

Marco Casini, Andrea Garulli, Antonio Giannitrapani, Antonio Vicino

*Dipartimento di Ingegneria dell'Informazione
Via Roma, 56 - 53100 Siena - ITALY
Email: {casini,garulli,giannitrapani,vicino}@ing.unisi.it*

Abstract: This paper presents an experimental setup for multi-robot systems based on the LEGO Mindstorms NXT technology. The team of mobile robots is supervised by a vision system, which allows one to simulate different types of sensors and communication architectures. The whole setup is embedded in the Automatic Control Telelab (<http://act.dii.unisi.it>), a remote lab featuring several educational experiences in control. Remote users can design control laws for the multi-agent system in the Matlab environment and test them by performing real experiments in the proposed setup. The paper presents some experiments showing how this remote lab can stimulate students' interest in mobile robotics.

Keywords: remote labs, mobile robotics, LEGO Mindstorms, multi-robot systems

Figur 7.2: Artikkel presentert ved konferanse.

7.4 Beskrivelse av litteratur

Siden denne artikkelen bare hadde 4 forfattere, får den *tag'en* CasGarGiaVic2011 i .bib-fila, se under

```
@InProceedings{CasGarGiaVic2011,  
    author = {Marco Casini and Andrea Garulli and Antonio Giannitrapani and Antonio Vicino},  
    title = {A LEGO Mindstorms multi-robot setup in the Automatic Control Telelab},  
    booktitle = {Proceedings of the 18th World Congress},  
    year = {2011},  
    pages = {9812--9817},  
    organization = {IFAC},  
}
```

Den fullstendige beskrivelsen av InProceedings er:

```
@InProceedings{,  
    author = {},  
    title = {},  
    OPTcrossref = {},  
    OPTkey = {},  
    OPTbooktitle = {},  
    OPTyear = {},  
    OPTeditor = {},  
    OPTvolume = {},  
    OPTnumber = {},  
    OPTseries = {},  
    OPTpages = {},  
    OPTmonth = {},  
    OPTaddress = {},  
    OPTorganization = {},  
    OPTpublisher = {},  
    OPTnote = {},  
    OPTannote = {}  
}
```

7.4 Beskrivelse av litteratur

7.4.6 `@PhDThesis` og `@MastersThesis`

Dersom du skal referere til andre studenters forskningsarbeid, kan du bruke `@PhDThesis` og `@MastersThesis`. Som du ser under nedenfor, benytter disse identiske elementer. Det finnes også en `@Thesis` som kan brukes for bacheloroppgaver.

```
@PhdThesis{,
    author = {},
    title = {},
    school = {},
    year = {},
    OPTkey = {},
    OPTtype = {},
    OPTaddress = {},
    OPTmonth = {},
    OPTnote = {},
    OPTannote = {}
}
```

```
@MastersThesis{,
    author = {},
    title = {},
    school = {},
    year = {},
    OPTkey = {},
    OPTtype = {},
    OPTaddress = {},
    OPTmonth = {},
    OPTnote = {},
    OPTannote = {}
}
```

7.5 Referering til litteratur, `\cite{ }`

7.5 Referering til litteratur, `\cite{ }`

For å referere til de forskjellige litteraturelementene i `referanser.bib`, må du bruke `\cite`-kommandoen som vist i følgende eksempel.

```
Som vist i \cite{Dre2019} kan numerisk integrasjon benyttes til ...
```

og denne vil fremstå slik:

Som vist i [3] kan numerisk integrasjon benyttes til ...

Det er vanlig at referanselisten kommer etter konklusjonen. For å få dette til må du inkludere følgende kodelinjer i hovedfilen rett etter konklusjonskapittelet:

```
\bibliographystyle{plain}
\bibliography{referanser.bib}
\addcontentsline{toc}{chapter}{Bibliografi}
```

Første linje forteller hvilken stil du ønsker å bruke. Stilen som heter `plain` plasserer et tall foran hver referanse i referanselista, og det er dette tallet som vises i teksten.

Andre linje forteller L^AT_EX hvilken .bib-fila som skal benyttes. Her kan det stå flere filnavn.

Siste linje legger til ”Bibliografi”-kapittelet i innholdsfortegnelsen.

Kapittel 8

Litt om ligninger

8.1 Bruk av `equation`-miljøet

For å skrive ligninger i L^AT_EX brukes `equation`-miljøet på følgende måte:

```
\begin{equation}
\label{eq:sinus}
y(t)=\sin(\omega t)
\end{equation}
```

som resulterer i følgende ligning hvor ligningsnummeret (8.1) kommer automatisk opp på høyre side av ligningen.

$$y(t) = \sin(\omega t) \tag{8.1}$$

8.2 Bruk av `align`-miljøet

For å sette flere ligninger over hverandre er `align` nyttig. Denne benytter tegnet & for å markere hvilken plass i ligningene som skal *alignes*. I eksemplet under er det likhetstegnet og `\downarrow` som skal stå over hverandre. For hver linjeslutt (**utenom den siste linjen**) må du indikere det med `\backslash\`.

8.3 Bruk av matematikkmodus i rapporttekst, \$x\$

```
\begin{align}
y(t) &= \sin(\omega t) \tag{\ref{eq:sinus}} \\
&\quad \& \downarrow \notag \\
y(t) &= \cos \Bigl( \omega t - \frac{\pi}{2} \Bigr)
\label{eq:cosinus}
\end{align}
```

Resultatet av koden er vist under

$$y(t) = \sin(\omega t) \tag{8.1}$$

↓

$$y(t) = \cos\left(\omega t - \frac{\pi}{2}\right) \tag{8.2}$$

Noen kommentarer til resultatet:

- Legg merke til at ligningsnummeret (8.1) til den øverste ligningen er det samme i ligningen i forrige delkapittel, og dette er gjort ved å bruke `\tag{}`-funksjonen som manuelt *tag'er* en ligning med ligningsnummeret til en annen ligning ved bruk av `\ref{}`-funksjonen fra kapittel 7.3. Denne måten å lage ligningsnummer på brukes bare når ligningene er identiske, slik som de er i dette tilfelle.
- Videre ser du at `\downarrow`-pilen ikke har ligningsnummer (som er jo logisk), og dette løses med `\notag`-funksjonen.
- Legg også merke til at selve sinus- og cosinusfunksjonene i ligningene (8.1) og (8.2) ser ut som vanlig stående tekst siden de skrives med kommandoene `\sin` og `\cos`.
- Legg merke til at siste linje IKKE skal ha `\` på siste linje. Dersom du glemmer dette, vil det dukke opp en tom linje med ligningsnummer.

8.3 Bruk av matematikkmodus i rapporttekst, \$x\$

Når du i teksten skriver om variabler som brukes i ligningene må du huske å skrive disse i matematikkmodus, det vil si med dollartegn foran og etter symbolet. I ligning (8.1) brukes for eksempel tiden t , hvor L^AT_EX-koden for å skrive tidsvariabel i matematikkmodus er `t`.

8.4 Bruk av vanlig tekst i ligninger, $\text{\textit{}} \text{\texttt{\textbackslash mathrm\{ }} \text{\texttt{\textbackslash mathit\{ }}$

Unngå å skrive ligninger i selve teksten. Dette fordi det ofte er vanskelig å lese, og du kan heller ikke referere til ligningen senere i teksten. Bruk derfor $\begin{equation}$ og $\end{equation}$ for alle ligninger.

8.4 Bruk av vanlig tekst i ligninger, $\text{\textit{}} \text{\texttt{\textbackslash mathrm\{ }} \text{\texttt{\textbackslash mathit\{ }}$

Det er ikke ofte du har bruk for å skrive vanlig tekst i ligningene, men skal du gjøre dette kan du bruke $\text{\textit{}}$ eller $\text{\texttt{\textbackslash mathrm\{ }}$ kommandoene. Et eksempel på dette er vist i ligning (8.3), hvor $\text{\texttt{\textbackslash underbrace\{ }}$ -kommandoen og $\text{\texttt{\textbackslash hspace*\{ }}$ -kommandoen er brukt i $\text{\texttt{\textbackslash align}}$ -miljøet.

```
\begin{align}
y(t) &= \underbrace{\sin(\omega t)}_{\text{Roman font}}
&\quad + \underbrace{\cos(2 \pi f t)}_{\text{Italic font}} \label{eq:sin_og_cos} \\
&& \hspace*{5mm} \text{\textit{}} \text{\texttt{\textbackslash mathrm\{Roman font\}}}
&& \hspace*{7mm} \text{\textit{}} \text{\texttt{\textbackslash mathit\{Italic\~font\}}}\text{\texttt{\textbackslash notag}}
\end{align}
```

som gir følgende resultat (legg merke til at du må bruke tilde for å lage mellomrom)

$$y(t) = \underbrace{\sin(\omega t)}_{\text{Romanfont}} + \underbrace{\cos(2\pi ft)}_{\text{Italic font}} \tag{8.3}$$

8.5 Bruk av $\text{\texttt{\textbackslash hspace*\{ }}$

I eksempelet over ble kommandoen $\text{\texttt{\textbackslash hspace*\{ }}$ brukt. Denne betyr *horizontal space*, og tilsvarende finnes det også $\text{\texttt{\textbackslash vspace*\{ }}$ betyr *vertical space*. For at ikke L^AT_EX skal ignorere dine ønsker å horisontal og vertikal avstand, MÅ du bruke * slik som vist i koden over.

Husk at du kan flytte både til høyre/venstre og ned/opp ved å bruke hhv. positive og negative verdier, og du kan angi forflytningen i mm, cm, eller punkter pt.

Kapittel 9

Litt om figurer

9.1 Bruk av `figure`-miljøet

For å inkludere figurer bruker du `figure`-miljøet på følgende måte

```
\begin{figure}[H]
    \centering
    \scalebox{0.6}{\includegraphics{figurfilnavn}}
    \caption{Figurtekst som beskriver hva vi ser i figuren.}
    \label{fig:xxxxx}
\end{figure}
```

Den store H-en etter `\begin{figure}` betyr her, eller *here* på engelsk. Det betyr at figuren plasseres akkurat der den står i teksten. For å bruke `[H]` må du inkludere pakken `here` på følgende måte `\usepackage{here}`.

Dersom du vil at \LaTeX skal bestemme plasseringen av figuren ved kompilering, kan du f.eks. skrive `[ht]` som betyr at du ønsker at den skal plasseres her og helst mot toppen øverst på en side. Ved å spesifisere slik havner ofte figuren en helt annen plass i rapporten, og det er litt slitsomt.

Kommandoen `\scalebox{0.6}` skalerer figuren til 60% av opprinnelig størrelse.

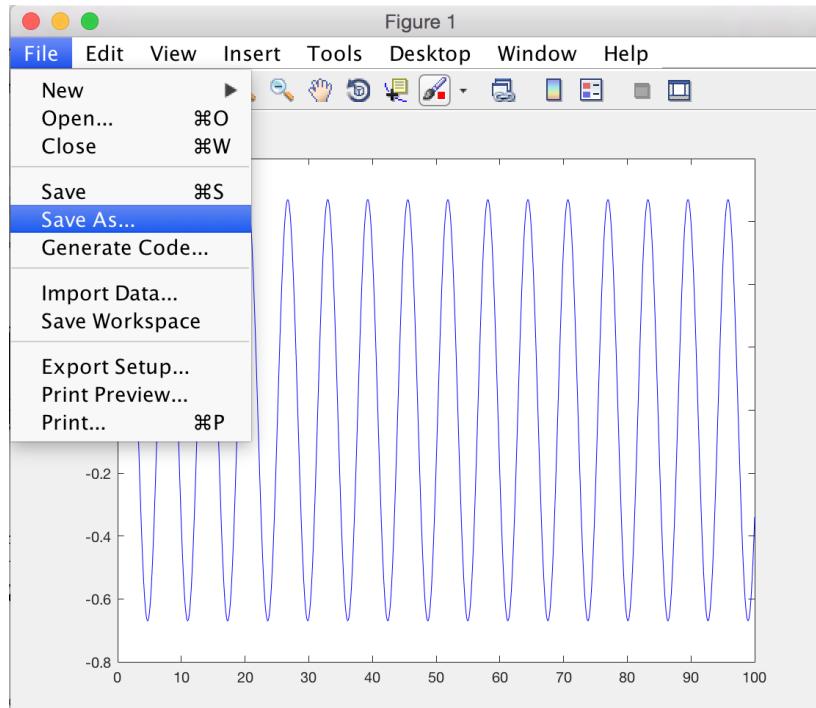
9.2 MATLAB-figurer

For oversiktens skyld pleier jeg ofte å la `xxxxx` i figur-tag'en `fig:xxxxx` være det samme som navnet på figurfila, dvs `\label{fig:figurfilnavn}`

9.2 MATLAB-figurer

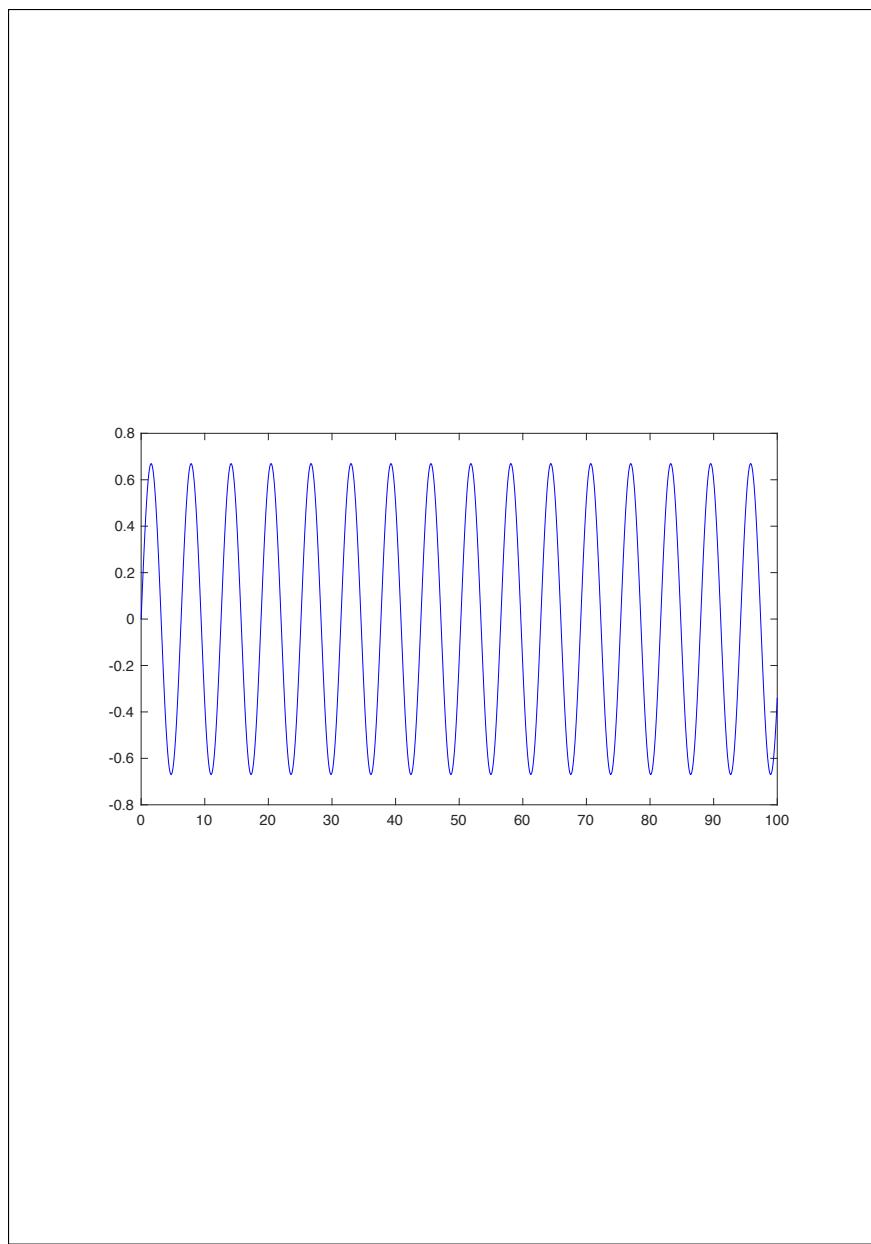
9.2.1 Format og størrelse

For at MATLAB-figurer skal være lesbar i rapporten må du bruke `png`- eller `pdf`-formatet. Bruker du `jpg` vil detaljene smøres utover og det er ofte umulig å lese ut informasjon av slike figurer. **Inntrykket av rapporten blir dermed dårlig.** Dersom du lagrer en MATLAB-figur til pdf-formatet ved å bruke `File`-menyen som vist i figur 9.1, vil det komme mye tomrom med i figuren som vist i figur 9.2.



Figur 9.1: Dersom du lagrer figuren i pdf-format via denne menyen, blir det mye hvitt rundt figuren, se figur 9.2.

9.2 MATLAB-furer



Figur 9.2: Dårlig figur med mye hvitt tomrom. Det er lagt på en ramme for å vise hvor mye hvitt tomrom som kommer med.

Dette er med andre ord ikke måten å lage pdf-furer på.

9.2 MATLAB-figurer

9.2.2 Funksjonen `SaveMyFigure.m`

For at du effektivt skal lage pdf-figurer uten dette store hvite feltet, finner du i mappen `MineFunksjoner` en funksjon som heter `SaveMyFigure.m`. Denne kaller du med følgende kode i *Command Window* (husk å stå i den mappen hvor du ønsker figuren lagret):

```
SaveMyFigure(gcf,'figurnavn')
```

hvor `gcf` står for `get current window`, som betyr at funksjonen lager figur av den MATLAB-figuren du sist trykket med musepekeren i. Funksjonen lager både en .pdf-versjon og en .fig-versjon, hvor .fig-versjonen kan åpnes i ettertid slik at du ved en senere anledning kan endre/legge til ting i figuren. Koden for funksjonen er i sin helhet gjengitt i kodeutdrag 9.1. På side 68 vises det hvordan du kan inkludere kode på denne måten.

Kode 9.1: Funksjonen `SaveMyFigure.m`

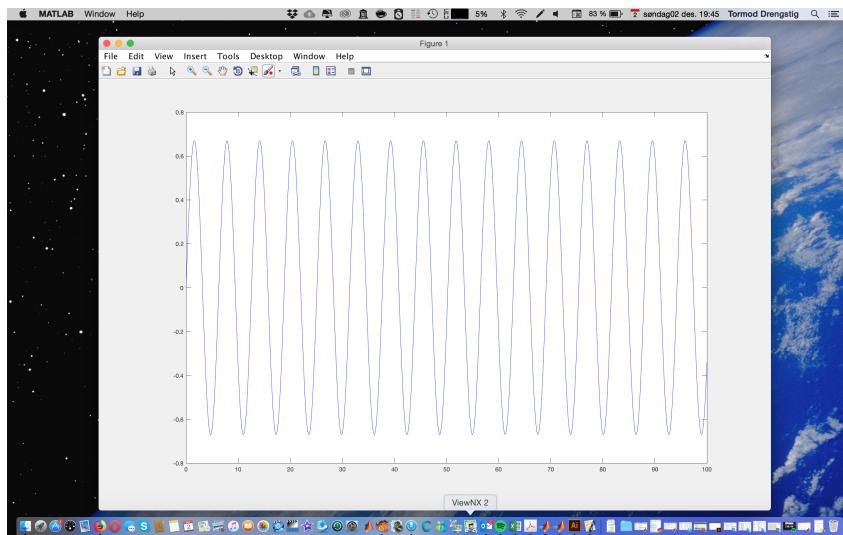
```
1 function [] = SaveMyFigure(fig_handle,name)
2
3 % SaveMyFigure, lagrer .fig og .pdf av gjeldende figur
4 % Klikk på den figuren du vil lagre, og bruk funksjonen slik:
5 %
6 %   SaveMyFigure(gcf,'figurename')
7 %
8 % hvor 'figurename' er et beskrivende filnavn for figuren.
9
10
11 % Ved lagring til vektorgrafikk pdf
12 % må figurstørrelsen settes. Dette trengs ikke for bitmap
13 fig_handle.PaperPositionMode = 'auto';
14 figure_pos = fig_handle.PaperPosition;
15 fig_handle.PaperSize = [figure_pos(3) figure_pos(4)];
16
17 % lagre 2 varianter, en .fig slik at du kan endre på figuren
18 % senere, og en .pdf for rapport
19 figurename_1 = [name, '.fig'];
20 figurename_2 = [name, '.pdf'];
21
22 if ~exist(figurename_1)
23     savefig(figurename_1)
24     print('-dpdf',' -painters ',' -bestfit ',figurename_2)
25 else
26     TekstStreng = ['Filens ',figurename_1,...
```

9.2 MATLAB-figurer

```
27      ''' finnes fra f?r. Overskrive?];
28 svar=questdlg(TekstStreng,'Advarslel','Ja','Nei','Nei');
29 switch svar
30     case 'Ja'
31         savefig(figurename_1)
32         print('-dpdf','-painters','-bestfit',figurename_2)
33     case 'Nei'
34         return
35 end
36 end
37
38 end
```

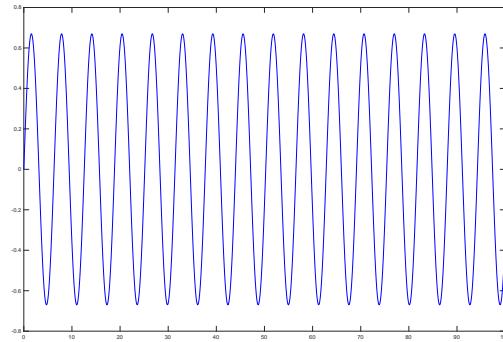
9.2.3 Effekten av vindusstørrelse før lagring

Dersom størrelsen på figuren i forhold til skjermstørrelsen er stor slik som vist i figur 9.3, vil resultatet etter lagring med funksjonen SaveMyFigure bli som i figur 9.4 hvor du er at tallene langs x- og y-aksen i figur 9.4 nesten uleselige. Dette skyldes at MATLAB-figuren i utgangspunktet dekket nesten hele skjermen som vist i figur 9.3.



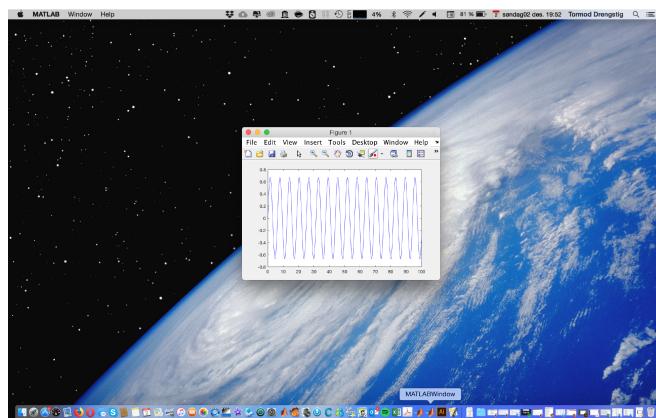
Figur 9.3: MATLAB-figuren fyller ut nesten hele skjermen.

9.2 MATLAB-figurer



Figur 9.4: Slik blir pdf-versjonen av MATLAB-figuren i figur 9.3 når lagres med SaveMyFigure-funksjonen. Tekstfonten er altfor liten, og tallverdiene på aksene er uleselig.

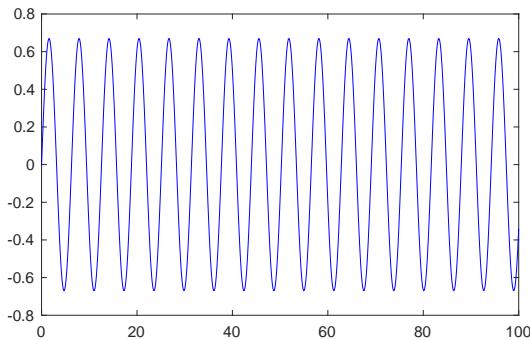
Dersom du reduserer størrelsen på selve MATLAB-figurvinduet som vist i figur 9.5 før du lagrer, vil fontstørrelsen på aksene fremstå som relativt mye større i forhold til selve figuren.



Figur 9.5: MATLAB-figuren er redusert i størrelse før lagring med SaveMyFigure-funksjonen.

Konsekvensen av dette er at leseren faktisk er i stand til å se tallene, se figur 9.6.

9.2 MATLAB-figurer



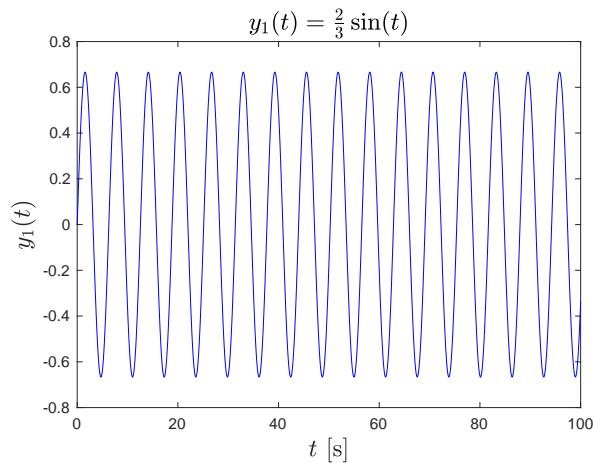
Figur 9.6: MATLAB-figuren i figur 9.5 lagret med `SaveMyFigure`-funksjonen. Legg merke til den fornuftige fontstørrelsen på x- og y-aksen.

9.2.4 Inkludering av \LaTeX -tekst i MATLAB-figurene

En figur skal alltid ha tekstbasert indikering av hva som vises på x- og y-aksene. I figurene i forrige delkapittel var dette helt fraværende, og derfor vises det nå hvordan du kan få \LaTeX -font på tekststrenge. Utgangspunktet er koden for selve sinuskurven i figur 9.6, men hvor det er lagt til `title`, `xlabel` og `ylabel` med \LaTeX -font og fontstørrelse 16.

```
t=0:0.1:100;
y_1=2/3*sin(t);
plot(t,y_1)
title('$y_1(t)=\frac{2}{3}\sin(t)$','Interpreter','latex');
xlabel('$t$ [s]','interpreter','latex')
ylabel('$y_1(t)$','interpreter','latex')
```

9.2 MATLAB-figurer



Figur 9.7: Eksempel på en god MATLAB-figur med tydelige, lesbare akseverdier og tekst på x- og y-akse, med figurittel.

Kapittel 10

Litt om tabeller

10.1 Bruk av `table`- og `tabular`-miljøene

Tabeller kan være vanskelig i L^AT_EX. Under er det vist noen eksempler på tabeller slik at du kan bruke de som et utgangspunkt for dine egne tabeller. Husk at tabelltekst skal stå over tabellen, imotsetning til figurtekst som står under.

Følgende kode gir tabell 10.1 som resultat.

```

\begin{table}[H]
\centering
\caption{Enkel tabell.}
\begin{tabular}{|l|r c|r|c|}\hline
    Venstrejustert & Høyrejustert & Sentrert \\
    & Høyrejustert & Sentrert \\ \hline\hline
    1.3434 & 0.55677 & 0.2 & 1.45 & 0.3 \\
    2.5667 & 0.66 & 0.4 & 100.01 & 1.43435 \\ \hline
\end{tabular}
\label{tab:enkel_tab}
\end{table}

```

10.1 Bruk av `table`- og `tabular`-miljøene

Tabell 10.1: Enkel tabell.

| Venstrejustert | Høyrejustert | Sentrert | Høyrejustert | Sentrert |
|----------------|--------------|----------|--------------|----------|
| 1.3434 | 0.55677 | 0.2 | 1.45 | 0.3 |
| 2.5667 | 0.66 | 0.4 | 100.01 | 1.43435 |

Hvert element i tabellen skiller med et &-tegn, og det kan være nyttig å organisere disse &-tegnene oppå hverandre slik at tabellen er lettere å lese i .tex-fila.

Detaljene som er vist i følgende linje i koden over

```
\begin{tabular}{|l|r c|r|c|}\hline
```

betyr at tabellen skal inneholde 5 kolonner hvor

- symbollet `|` betyr loddrett tabellstrek,
- `l` betyr *left* og dermed venstrejustert
- `r` betyr *right* og dermed høyrejustert
- `c` betyr *center* og dermed sentrert
- kommanoden `\hline` betyr *horizontal line* og dermed vannrett tabellstrek

10.1 Bruk av `table`- og `tabular`-miljøene

Koden for en litt mer avansert tabell er vist under og i tabell 10.2. Denne inneholder både `\multirow`, `\multicolumn`, `\cellcolor`, og `\cline`.

```
\begin{table}[H]
\centering
\renewcommand{\arraystretch}{1.2}
\caption{Parametre brukt i simuleringen.}
\begin{tabular}{|c||c|c||c|c|c|}\hline
\multirow{2}{*}{Controller} & \multicolumn{2}{c||}{Variables} & \multicolumn{3}{c}{Parameters} \\
\cline{2-6}
no. & \cellcolor[gray]{0.8} $h_{high}$ & $u_{high}$ & $k_s^u$ & $V_{max}^u$ & $K_M^u$ \\
& \cellcolor[gray]{0.8} 0.55 & 0.49 & 0.01 & 0.0071 & 0.137 \\
1 & \cellcolor[gray]{0.8} 0.66 & 0.47 & 0.01 & 0.0177 & 0.797 \\ \hline
\end{tabular}
\label{tab:avansert_tab}
\end{table}
```

Tabell 10.2: Parametre brukt i simuleringen.

| Controller no. | Variables | | Parameters | | |
|-------------------|------------|------------|------------|-------------|---------|
| | h_{high} | u_{high} | k_s^u | V_{max}^u | K_M^u |
| 1 | 0.55 | 0.49 | 0.01 | 0.0071 | 0.137 |
| 2 | 0.66 | 0.47 | 0.01 | 0.0177 | 0.797 |

Legg merke til at kommandoen `\renewcommand{\arraystretch}{1.2}` øker høyden med 20% for hver rekke i tabellen. Legg også merke til at du kan flytte neste linje opp eller ned med å legge skrive en høyde i hakparentes, f.eks. `[-4mm]`, bak linje-skiftkommandoen `\``.

Kapittel 11

Litt om punktlister

Punktlister kan du lage med `\itemize`- og `\enumerate`-kommandoene.

11.1 Bruk av `\itemize`

For å styre avstanden som er mellom punktene i en liste bruker du `\setlength\itemsep{}`-kommandoen som du plasserer inne i mellom `\begin{itemize}` og `\end{itemize}`.
Følgende kode

```
\begin{itemize}
  \setlength\itemsep{0mm}
  \item Dette er
  \item en punktliste
  \begin{itemize}
    \setlength\itemsep{-2mm}
    \item som går inn
    \item flere nivå
  \end{itemize}
  \item før den er tilbake igjen
\end{itemize}
```

11.1 Bruk av `\itemize`

gir denne punktlisten:

- Dette er
- en punktliste
 - som går inn
 - flere nivå
- før den er tilbake igjen

Du kan også velge helt selv hva du vil at punktnummereringen skal være. For eksempel denne selvvalgte nummereringen

```
\begin{itemize}
\item [3)] På tredje plass ...
\item [2)] På andre plass ....
\item [1)] På første plass ...
\end{itemize}
```

gir denne listen

- 3) På tredje plass ...
- 2) På andre plass
- 1) På første plass ...

11.2 Bruk av `\enumerate`

11.2 Bruk av `\enumerate`

Listen `enumerate` kan brukes på mange måter.

11.2.1 Tallbasert liste, standardversjonen

Ved å bruke `enumerate` på følgende måte

```
\begin{enumerate}
\item Dette er en standard
\item tallbasert liste
\end{enumerate}
```

får vi denne listen

1. Dette er en standard
2. tallbasert liste

11.2.2 Bokstavbasert liste

Enumerate-listen kan endre ved å spesifisere andre tellemåter i hakeparentes bak `\begin{\enumerate}`. Følgende kode

```
\begin{enumerate}[a]
\item Dette er en
\item bokstavbasert liste
\end{enumerate}
```

gir denne listen

- a) Dette er en
- b) bokstavbasert liste

11.2 Bruk av `\enumerate`

11.2.3 Romertallbasert liste

Alternativt kan vi bruke romertall som vist under. Følgende kode

```
\begin{enumerate}[i.]
\item Dette er en
\item romertall liste
\item
\item
\item
\end{enumerate}
```

gir denne liste

- i. Dette er en
- ii. romertall liste
- iii.
- iv.
- v.

Kapittel 12

Litt om inkludering av MATLAB-kode

12.1 Pakkene `listings` og `mcode`

Ved å legge til pakken `listings` kan du enkelt inkludere all slags kode i \LaTeX . Pakken `mcode` er en tilleggspakke som tilpasser `listings` til å presentere MATLAB-kode.

Du kan typisk bruke `listings` på 2 måter som vist i de neste delkapitlene.

12.1.1 Dynamisk med `\lstinputlisting`-kommandoen

Ved å bruke `\lstinputlisting`-kommandoen oppretter du en dynamisk link til selve fila du ønsker å inkludere i rapporten. Fordelen med å inkludere kode på denne måten er at dersom du oppdaterer MATLAB-fila, vil rapporten oppdateres automatisk neste gang du kompilerer \LaTeX -dokumentet.

12.1 Pakkene `listings` og `mcode`

Et eksempel på hvor slik kode er brukt er i kodeutdrag 9.1, hvor L^AT_EX-koden ser slik ut:

```
\lstinputlisting[caption={Funksjonen {\tt SaveMyFigure.m}.},  
label=kode:SaveMyFigure]{SaveMyFigure.m}
```

Ved å ytterligere spesifisere linjenummer kan du velge å presentere kun deler av koden, se under.

```
\lstinputlisting[firstnumber=14,firstline=14,lastline=28,  
caption={Utdrag av funksjonen {\tt SaveMyFigure.m}.},  
label=kode:Utdrag_av_SaveMyFigure]{SaveMyFigure.m}
```

Resultatet av denne koden blir som følger:

Kode 12.1: Utdrag av funksjonen `SaveMyFigure.m`.

```
14 figure_pos = fig_handle.PaperPosition;  
15 fig_handle.PaperSize = [figure_pos(3) figure_pos(4)];  
16  
17 % lagre 2 varianter, en .fig slik at du kan endre p? figuren  
18 % senere, og en .pdf for rapport  
19 figurename_1 = [name,'.fig'];  
20 figurename_2 = [name,'.pdf'];  
21  
22 if ~exist(figurename_1)  
23     savefig(figurename_1)  
24     print('-dpdf','-painters','-bestfit',figurename_2)  
25 else  
26     TekstStreng = ['Filten ''',figurename_1,...  
27         ''' finnes fra f?r. Overskrive?'];  
28     svar=questdlg(TekstStreng,'Advarslet','Ja','Nei','Nei');
```

12.1 Pakkene `listings` og `mcode`

12.1.2 Statisk med `\lstlisting`-kommandoen

Ved å bruke `\lstlisting`-kommandoen inkluderer du hele koden eller deler av koden i selve .tex-fila.

Ulempen er at du må oppdatere koden i rapporten dersom du gjør endringer i koden i MATLAB. Denne måten klarer heller ikke å gjengi bokstavene æ, ø og å som du bruker i kommentarer, men dette er ikke et stort problem.

Et eksempel på slik kodegjengivelse er kodeutdrag 6.10 som er laget på følgende måte:

```
\begin{lstlisting}[caption={Kode for turtallsregulator for
    motor A.}, label=kode:turtall, firstnumber=11]
% trykknapp bestemmer om regulator er i auto eller manuel

Ref_TurtallA(k) = PotMeter(k)
e_TurtallA(k) = Ref_TurtallA(k) - TurtallA(k-1);
P_A(k) = Kp*e_TurtallA(k);
I_A(k) = EulerForward(I_A(k-1), Ki*e_TurtallA(k-1), Ts(k-1));
e_TurtallA_IIR(k) = IIR_filter(e_TurtallA_IIR(k-1), e_TurtallA(k), alfa2);
D_A(k) = Derivation(Kd*e_TurtallA_IIR(k-1:k), Ts(k-1));

TurtallsReg(k) = ~Bryter(k);
if TurtallsReg(k)
    PowerA(k) = P_A(k) + I_A(k) + D_A(k);
else
    PowerA(k) = PotMeter(k);
    P_A(k) = NaN;
    I_A(k) = PowerA(k);
    D_A(k) = NaN;
end

motorA.Speed = PowerA(k);
start(motorA);
\end{lstlisting}
```

12.1 Pakkene `listings` og `mcode`

Koden gir følgende resultat:

Kode 12.2: Kode for turtallsregulator for motor A.

```
11 % trykknapp bestemmer om regulator er i auto eller manuel
12
13 Ref_TurtallA(k) = PotMeter(k)
14 e_TurtallA(k) = Ref_TurtallA(k) - TurtallA(k-1);
15 P_A(k) = Kp*e_TurtallA(k);
16 I_A(k) = EulerForward(I_A(k-1), Ki*e_TurtallA(k-1), Ts(k-1));
17 e_TurtallA_IIR(k) = IIR_filter(e_TurtallA_IIR(k-1), ...
    e_TurtallA(k), alfa2);
18 D_A(k) = Derivation(Kd*e_TurtallA_IIR(k-1:k), Ts(k-1));
19
20 TurtallsReg(k) = ~Bryter(k);
21 if TurtallsReg(k)
22     PowerA(k) = P_A(k) + I_A(k) + D_A(k);
23 else
24     PowerA(k) = PotMeter(k);
25     P_A(k) = NaN;
26     I_A(k) = PowerA(k);
27     D_A(k) = NaN;
28 end
29
30 motorA.Speed = PowerA(k);
31 start(motorA);
```

12.1.3 Pakken `figurepath`

For at den dynamiske versjonen med `lstinputlisting`-kommandoen skal finne .m-filene du spesifiserer, må du legge inn de forskjellige mappestiene hvor filene dine ligger i pakken `figurepath`. I denne rapporten er følgende mappestier spesifisert (se i hovedfilen `Gruppe19XX.tex`):

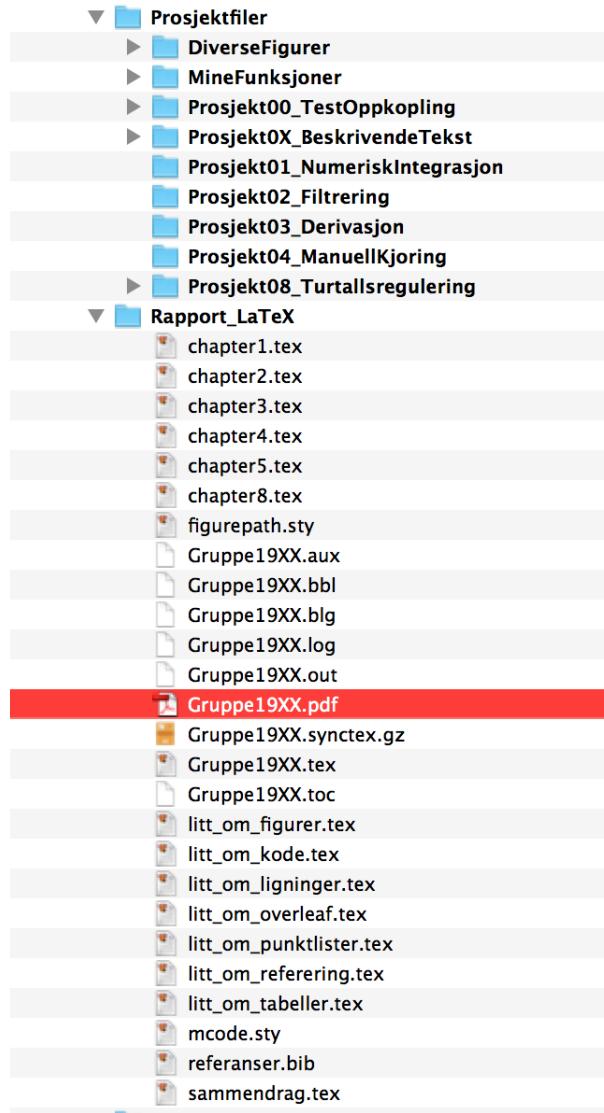
12.1 Pakkene `listings` og `mcode`

```
\usepackage[../Projektfiler/DiverseFigurer/,  
../Projektfiler/DiverseFigurer/overleaf/,  
../Projektfiler/MineFunksjoner/,  
../Projektfiler/Prosjekt01_NumeriskIntegrasjon/,  
../Projektfiler/Prosjekt02_Filtrering/,  
../Projektfiler/Prosjekt03_Derivasjon/,  
../Projektfiler/Prosjekt04_ManuellKjoring/,  
../Projektfiler/Prosjekt08_Turtallsregulering/] {figurepath}
```

Denne spesifiseringen tar utgangspunkt i at hovedfila `Gruppe19XX.tex` ligger under `Rapport_LaTeX`-mappen som vist i mappestrukturen i figur 12.1. Derfor brukes syntaxen `../ /` i figurepath-pakken for å gå opp ett nivå.

Som du også ser ligger pakken `figurepath.sty` i `Rapport_LaTeX`-mappen.

12.1 Pakkene `listings` og `mcode`

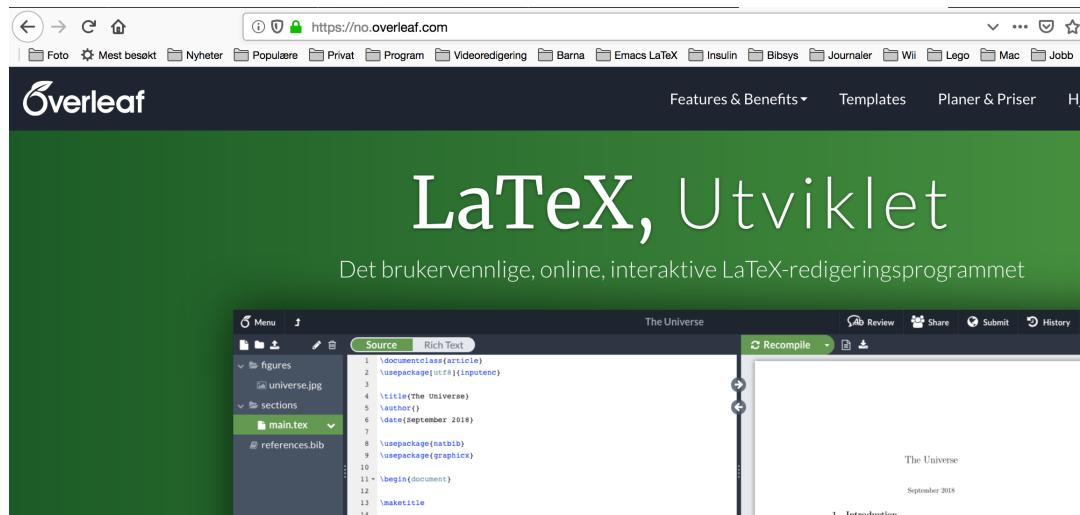


Figur 12.1: Organisering i mappestruktur hvor `DiverseFigurer`-mappen inneholder både figurer og undermapper med figurer, mens hovedfilen ligger i `Rapport_LaTeX`-mappen.

Kapittel 13

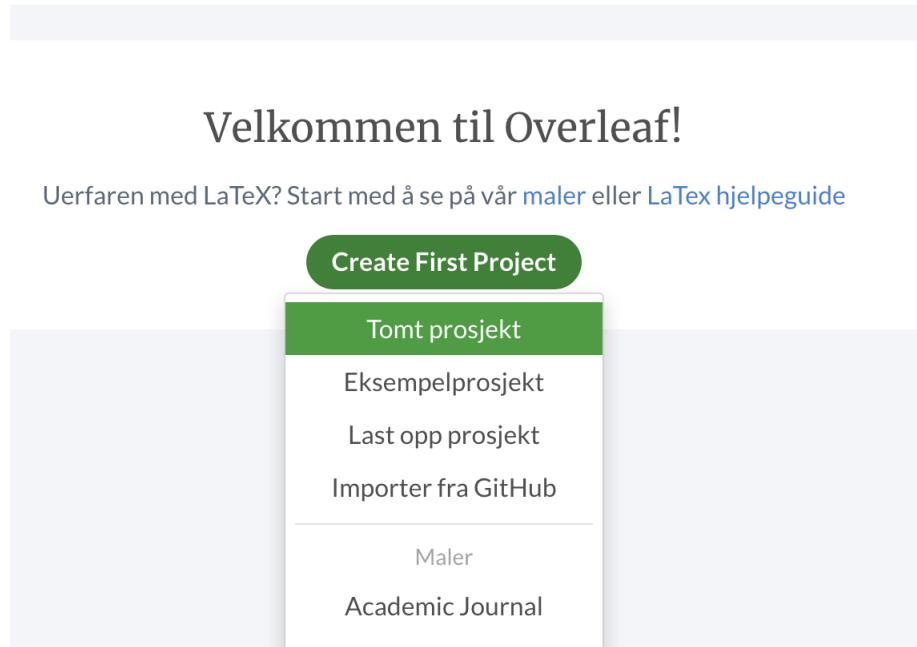
Litt om Overleaf

Dette kapittelet gir en introduksjon til Overleaf i form av en rekke skjermdump og beskrivende figurtekster.

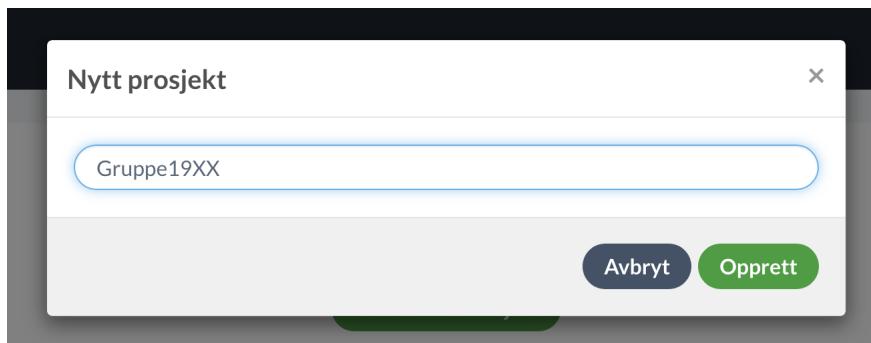


Figur 13.1: Gå inn på <https://no.overleaf.com/>. Registrer deg med UiS-epostadressen din.

Litt om Overleaf

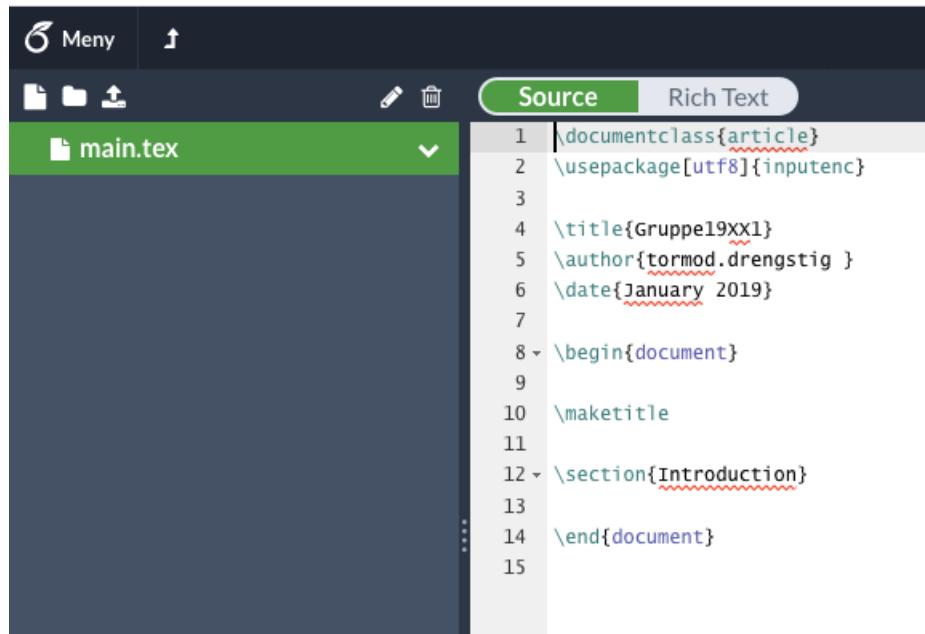


Figur 13.2: Etter at du har registrert deg og logget deg inn, trykk på “Create First Project” og velg “Tomt prosjekt”.



Figur 13.3: Kall prosjektet “Gruppe19XX” hvor XX er gruppernummeret deres.

Litt om Overleaf



The screenshot shows the Overleaf LaTeX editor interface. At the top, there's a menu icon and a 'Meny' button. Below the menu is a toolbar with icons for file operations like new, open, save, and upload. To the right of the toolbar are tabs for 'Source' (which is selected) and 'Rich Text'. A dropdown menu is open over the 'main.tex' file name. The code area contains the following LaTeX code:

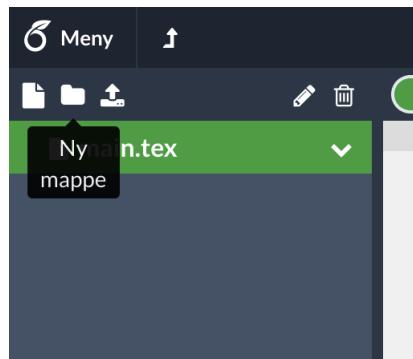
```
1 \documentclass{article}
2 \usepackage[utf8]{inputenc}
3
4 \title{Gruppe19XX1}
5 \author{tormod.drengstig }
6 \date{\textcolor{red}{January} 2019}
7
8 \begin{document}
9
10 \maketitle
11
12 \section{Introduction}
13
14 \end{document}
15
```

Figur 13.4: Det opprettes automatisk opp en hovedfil kalt `main.tex` i prosjektet.

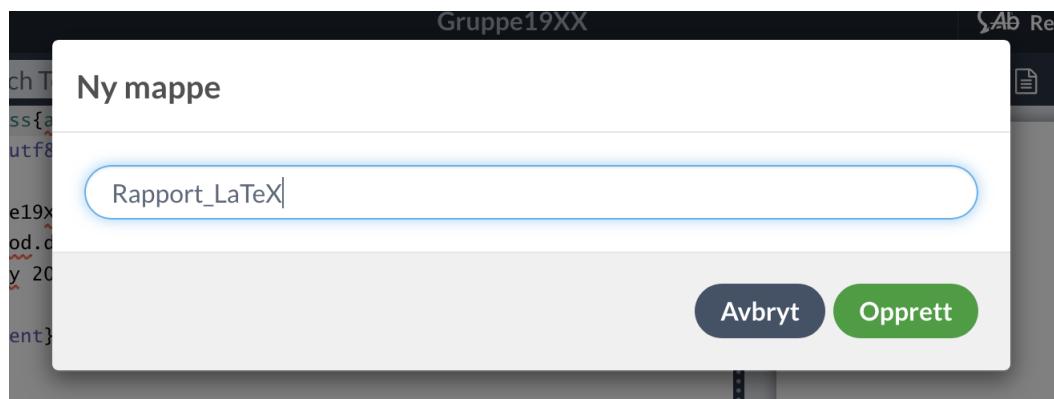


Figur 13.5: Nå er poenget at mappestrukturen som er å finne i `Lego.zip`-filen skal lages i Overleaf-prosjektet. Vær klar over at det i mappen `DiverseFigurer` ligger endel figurer samt `overleaf`-mappen.

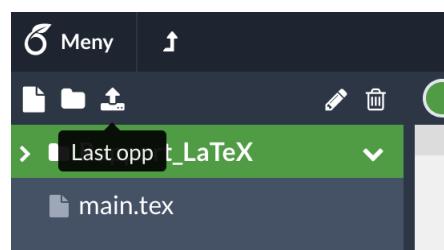
Litt om Overleaf



Figur 13.6: Trykk på symbolet "Ny mappe".

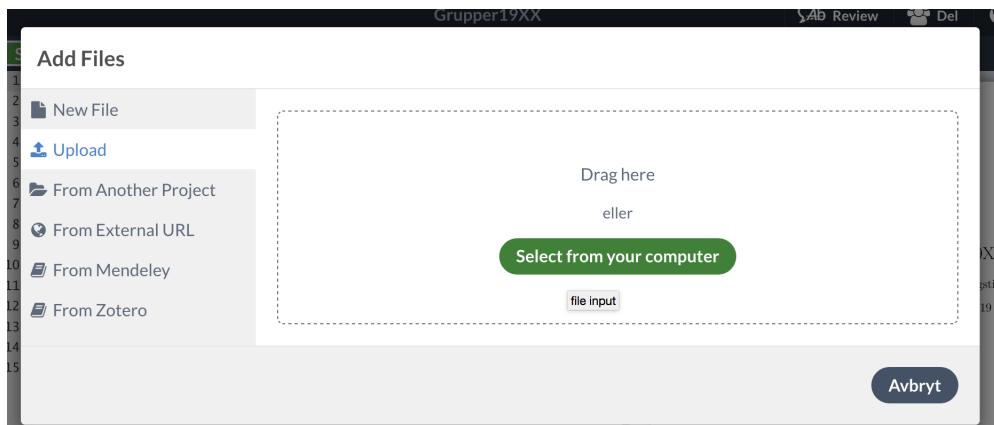


Figur 13.7: Lag mappen `Rapport_LaTeX`.

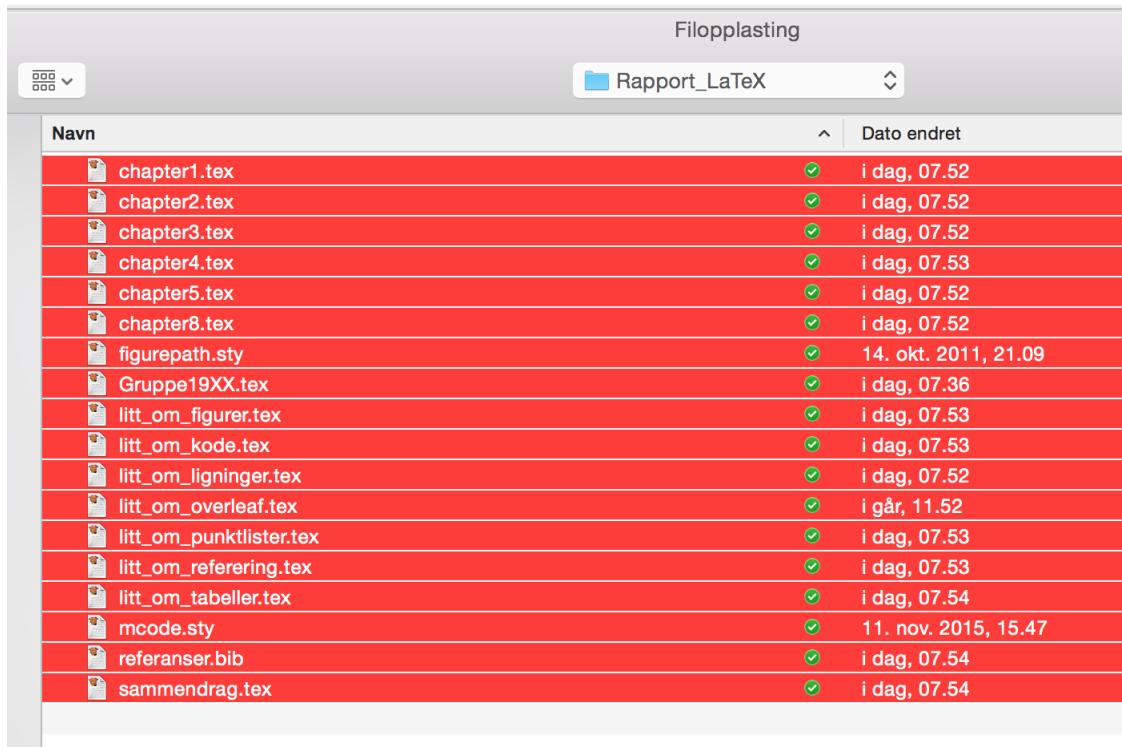


Figur 13.8: Stå i mappen `Rapport_LaTeX` og trykk "Last opp".

Litt om Overleaf

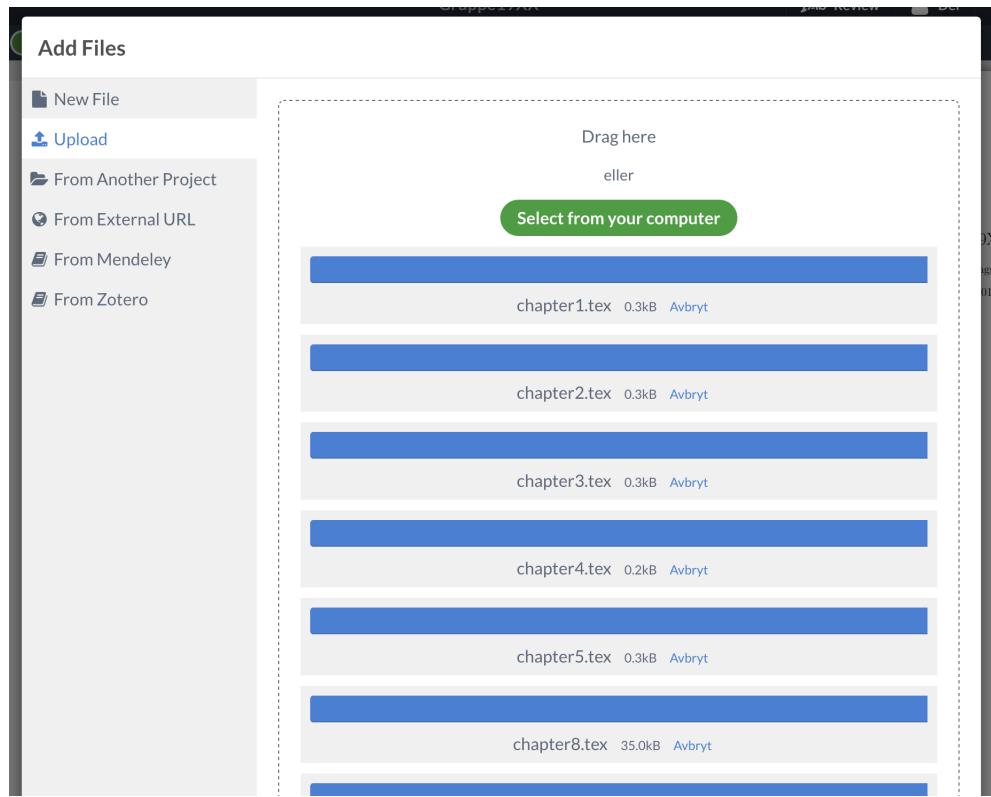


Figur 13.9: Trykk på "Select from your computer".

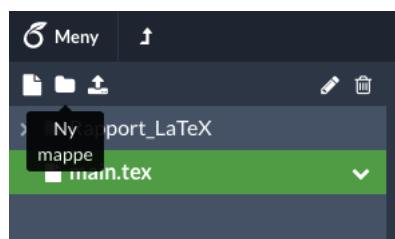


Figur 13.10: Naviger deg frem til mappen `Rapport_LaTeX` og velg alle filene.

Litt om Overleaf

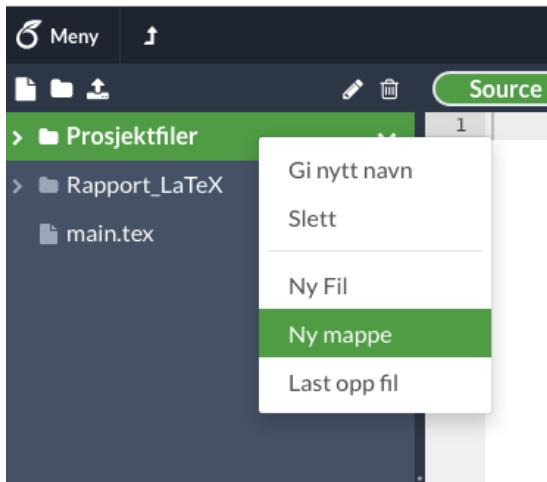


Figur 13.11: Filene lastes inn.

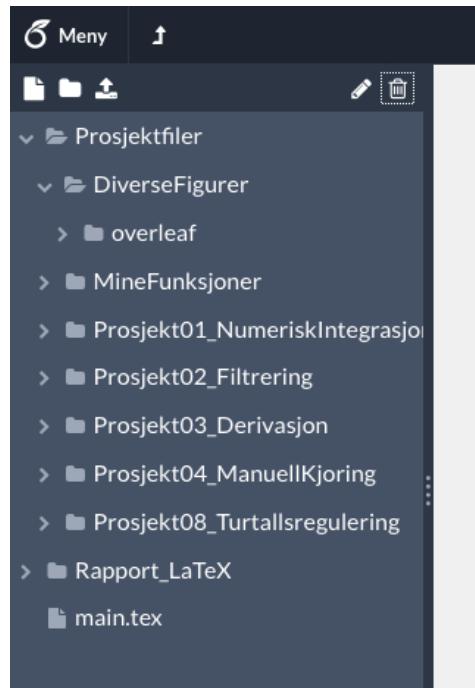


Figur 13.12: Stå i filen `main.tex`, og trykk "Ny mappe", og lag på samme måte som vist i figur 13.7 den nye mappen `Prosjektfiler`.

Litt om Overleaf

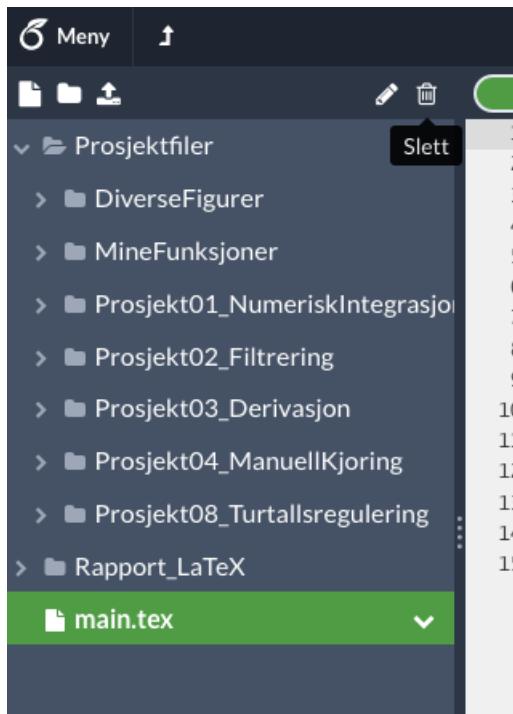


Figur 13.13: For å lage undermappene vist i figur 13.5, stå i mappen **Prosjektfiler**, høyreklikk, og velg “Ny mappe”. Lag på denne måten alle mappene som er vist i figur 13.14.

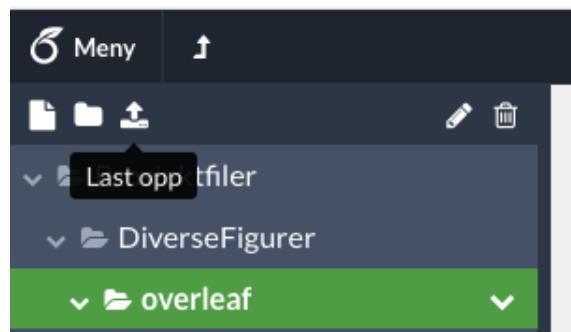


Figur 13.14: Mappestrukturen klar.

Litt om Overleaf



Figur 13.15: Slett filen `main.tex`.



Figur 13.16: Nå skal filene i hver mappe lastes inn på samme måte som vist i figurene 13.8 - 13.11. Start f.eks. med å stå i mappen `overleaf` og trykk "Last opp".



Figur 13.17: Alle filene er på plass. Legg merke til at filene i mappen `overleaf` ikke vises, at mappene Prosjekt01... - Prosjekt04... er tomme, samt at det kun er filen `SaveMFyFigure.m` som foreløpig ligger i mappen `MineFunksjoner`.

Litt om Overleaf

The screenshot shows the Overleaf LaTeX editor interface. On the left, there's a file browser with a sidebar for 'Prosjektfiler' and a main area for 'Rapport_LaTeX' containing files like chapter1.tex through chapter8.tex and figurepath.sty. A file named 'Gruppe19XX.tex' is selected and highlighted in green. A context menu is open over this file, with the option 'Gi nytt navn' (Give new name) highlighted in green. The main workspace on the right displays the LaTeX source code for 'Gruppe19XX.tex'. The code includes standard document class and package imports, along with specific Overleaf project path definitions and a listing environment setup.

```
\documentclass[11pt,a4]{report}
\usepackage[utf8]{inputenc}
\usepackage[T1]{fontenc}
\usepackage[norsk]{babel}
\renewcommand{\rmdefault}{cmss}
\usepackage{verbatim,amsmath}
\usepackage{enumerate}
\usepackage{fancyhdr,fancyvrb}
\usepackage{graphicx,color,boxedminipage}
\usepackage{ragged2e,colortbl,appendix}
\usepackage{here,hyperref,multirow,pdfpages}
\usepackage{xcolor}
\usepackage{lipsum,todonotes}

\usepackage[../Prosjektfiler/DiverseFigurer/,
../Prosjektfiler/DiverseFigurer/overleaf/,
../Prosjektfiler/MineFunksjoner/,
../Prosjektfiler/Prosjekt01_NumeriskIntegrasjon/,
../Prosjektfiler/Prosjekt02_Filtrering/,
../Prosjektfiler/Prosjekt03_Derivasjon/,
../Prosjektfiler/Prosjekt04_Manuellkjoring/,
../Prosjektfiler/Prosjekt08_Turtallsregulering/]{figurepath}

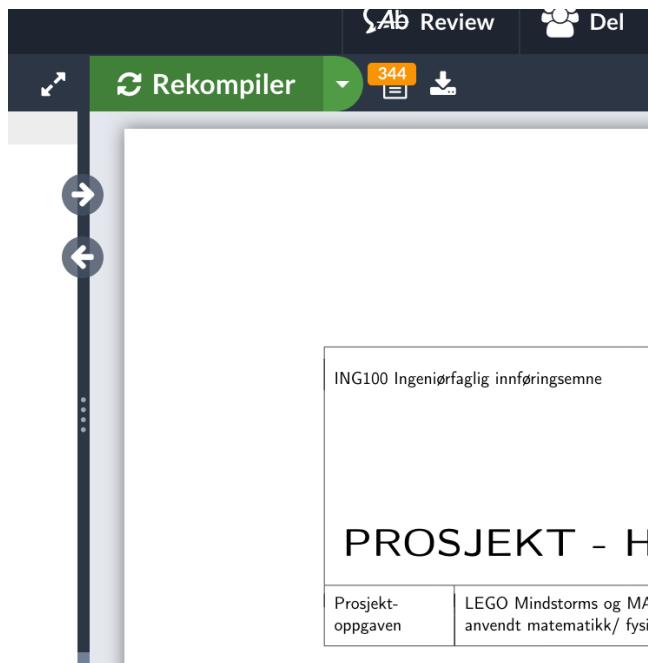
% inkludering av kode
\usepackage{listings}
\usepackage[framed,numbered,autolinebreaks,useliterate]{mcode}
% for å kunne skrive norske kommentarer i Matlab og vises i listingspakken
\lstset{inputencoding=ansinew}
```

Figur 13.18: Trykk på **Gruppe19XX.tex**. Høyreklikk på denne og endre navn til gruppenummeret deres.

Litt om Overleaf

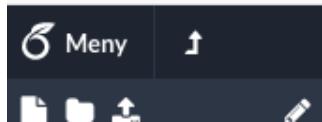


Figur 13.19: Trykk på knappen hvor det står "Rekompiler".



Figur 13.20: Etter en stund er den ferdig med å kompilere. Som du ser står det et tall med oransje bakgrunn. Det betyr at det er *warnings* i kompilering. Disse er ufarlige. Dersom bakgrunnen skifter farge til **rødt** er det kompileringsfeil, og disse må rettes opp. Trykk på selve tallet så kommer kompileringsresultatet opp og du kan se fra henvisning til linjenummer hvor feilen er.

Litt om Overleaf



Figur 13.21: For å legge til medstudenter i prosjektet, trykk først på pilen med knekk for å komme opp til prosjektsiden.

The screenshot shows the Overleaf website's header with several navigation options: 'Features & Benefits ▾', 'Templates', 'Hjelp ▾', 'Prosjekter', and 'Konto ▾'. A dropdown menu is open from the 'Konto' button, listing 'tormod.drengstig@uis.no', 'Kontoinnstillinger' (which is highlighted in green), 'Abonnement', and 'Logg ut'. Below the header, there is some descriptive text about institutions and collaboration.

Figur 13.22: Trykk deretter på "Konto" og "Kontoinnstillinger".

The screenshot shows the Overleaf website's header with the 'Features & Benefits ▾' button highlighted. A dropdown menu is open, listing 'For Writing', 'For Groups' (which is highlighted in green), 'For Teaching', 'For Universities', 'For Publishers', and 'For Enterprises'. Below the header, there is some descriptive text about institutions and collaboration.

Figur 13.23: Trykk på "Featres & Benefits" og så "For Groups".

Litt om Overleaf

The screenshot shows the Overleaf Groups account management interface. At the top, it says "Get Overleaf For Your Whole Group" and "(and don't worry about dealing with individual expense claims)". Below this, there's a section titled "Group Account" with a note "You have added 2 of 10 available members". A table lists two email addresses: "support@overleaf.com" and "john@overleaf.com". Both entries have a small "x" next to them under the "Accepted invite" column, indicating they have not been accepted. At the bottom, there's a button to "Add more members" and a field to enter email addresses like "jane@example.com, joe@example.com". There's also a green "Add" button and a "Export CSV" link.

| Email | Name | Accepted invite |
|---|------|-----------------|
| <input type="checkbox"/> support@overleaf.com | | x |
| <input type="checkbox"/> john@overleaf.com | | x |

Add more members

jane@example.com, joe@example.com

Add

Export CSV

Figur 13.24: Legg til studenter med sin registrerte epostadresse (UiS-adressen). Disclaimer: Jeg er noe usikker på dette her siden jeg ikke har hatt mulighet til å teste det ut.

Kapittel 14

Konklusjon

Konklusjonen skal generelt være en kort oppsummering av problemstilling, hva som er gjort og resultatene som er oppnådd.

Bibliografi

- [1] How can I use bibtex to cite a web page?
<https://tex.stackexchange.com/questions/3587/how-can-i-use-bibtex-to-cite-a-web-page>.
- [2] R. A. Adams and C. Essex. *Calculus. A complete course*. Pearson, 9 edition, 2017.
- [3] T. Drengstig. Lego mindstorms og matlab; anvendt matematikk og fysikk i skjønn forening. Technical report, Universitet i Stavanger, 2019. Utdelt materiale.

Vedlegg A

Timelister

Tabell A.1: Timelister for gruppe 19XX i faget ING100.

| Navn/uke | Per | Pål | Anne | Eva |
|----------|-----|-----|------|-----|
| 40 | | | | |
| 41 | | | | |
| 42 | | | | |
| 43 | | | | |
| 44 | | | | |
| 45 | | | | |
| 46 | | | | |
| Sum | | | | |

Vedlegg B

Programlisting prosjekt 04

B.1 P04_F4_MathCalculations.m

B.2 P04_F5_CalculateAndSetMotorPower.m

Vedlegg C

Programlisting prosjekt 05

C.1 P05_F4_MathCalculations.m

C.2 P05_F5_CalculateAndSetMotorPower.m