NTNU

Kunnskap for en bedre verden

Oppsett og videre arbeid

Dette dokumentet gir en rask innføring i hvordan MPC er satt opp, og hva som må/bør gjøres før den implementeres på Lone Wolf ATV. Videre følger en guide til oppsett og kjøring av Simulatoren.

1 MPC

MPCen har blitt utviklet med riktig ROS2 interface for at den skal være implementerbar med det reele systemet. Det er noen ting som må gjøres før MPC er klar til å kjøre på Lone Wolf, som vil bli oppsummert her.

Filene i MPCen er delt opp i to filer. Python filen for MPC, `MPC.py`, og MPC ROS noden, `MPC_node_main.py`. `MPC.py` har MPC koden, og er satt opp som en klasse. Denne inkluderes i Noden, hvor MPC objektet opprettes. Dette vises i kodesnutten under.

```
showing how the MPC object is created, label=fig:
    MPC_controller_made]
from MPC import MPC_Controller

class mpc_ros_controller(Node):
    def __init__(self):
        ...  # Code omitted for brevity
        self.mpc = MPC_Controller()
```

For å regne ut neste pådrag kan denne kommandoen brukes. `physical_states` er de 4 fysiske tilstandene X, Y, ψ, og v, og `xref_n` og `yref_n` er lister med X- og Y-koordinater.

```
u_t = self.mpc.mpc_generate_step(physical_states, xref_n, yref_n)
```

Hvis noe er uklart angående MPC og krav til videreutvikling før implementasjon på Lone Wolf kan Jørgen Vesterheim Knutsen kontaktes for oppklaring!

1.1 GPS koordinater

GPS koordinatene må konverteres til en lokal referanseramme. Det vil si at det må konverteres fra minutter, sekunder etc. til et kartesisk koordinatsystem, [X, Y], hvor Lone Wolf eller annet ønsket punkt er [0,0].

1.2 Extended Kalman Filter

Avhengig av hvor god data som hentes fra Sensorer og GPS kan det være nødvendig å implementere et Extended Kalman Filter. Vectornav er en høy-kvalitets IMU så sensordataen kan være god nok. I tillegg, hvis RTK i bruk vil GPS dataen også være svært god. Hvis det kun er vanlig GPS som brukes bør EKF implementeres.

1.3 Path planning

For testing kan en manuelt angitt referanse brukes. For full-autonom operasjon må path planning implementeres. Denne bør ta hensyn til hindringer, som trær, stup etc. Denne bør også forsøke å implementere hastighet i referansen, slik at en

trajectory er referansen istedenfor kun punkter. Hvis ikke trajectory blir implementert i path planning algoritmen, er det viktig at hasighet blir inbakt som en del av referansen.

1.4 Modell-tilpassning

Sjekk PDF for guide til hvordan tilpasse modellen til målt data! Dette bør gjøres før MPC testes slik at oppførselen blir mer forutsigbar.

1.5 Bremser

Den nåværende modellen har ikke implementert bremser. Dette er fordi den har en aggressiv av/ på karakteristikk. Dette bør måles data på før det implementeres. Dette BØR fikses før Lone Wolf slippes ut autonomt.

1.6 Elektronisk nødstopp

Fiks slik at den elektroniske nødstoppen setter på bremser. For øyeblikket kutter den bare gassen, og den ruller til den stopper. Dette kan lede til farlige situasjoner.

1.7 Bygge workspace på "Skapet"

Chat GPT er litt usikker på forskjellige ROS versjoner, som fort skaper forvirring. Anbefaler å bruke den offisielle guiden så mye som mulig her.

Det må bygges ROS pakker. Følgende link er en tutorial til hvordan man oppretter ROS pakker.

https:docs.ros.orgenfoxyTutorialsBeginner-Client-LibrariesCreating-Your-First-ROS2-Package.html

Dette må gjøres hvis koden skal kjøres av ROS ved å skrive "`ROS2 run pakkenavn kode`"

Det gjøres på følgende måte:

På Predator PCn kan man source ros ved å skrive "2" i kommoandovinduet. På skapet skal følgende gjøres:

```
source /opt/ros/foxy/setup.bash
```

Listing 1: Source ROS2

Bytt ut "ros2_ws" med passende navn.

```
mkdir -p ~/ros2_ws/src
cd ~/ros2_ws/src
```

Bygg pakken. bytt ut `--node-name my_node my_mypackage` med relevante navn

```
ros2 pkg create --build-type ament_python --node-name my_node
    my_package
```

Bygg pakken!

```
colcon build
```

Source ROS2. Dette kan varriere, og det kan være lurt å sjekke startup.sh for hvordan dette gjøres.

```
source install/local_setup.bash
```

Gå inn i pakken

```
cd ros2_ws pakkenavn pakkenavn
```

Gjør filene executable

```
sudo chmod +x filnavn
```

Filstrukturen bør se ut som dette

```
my_ros2_package/
          my_ros2_package/
                __init__.py
                MPC_node_main.py
                MPC.py
          setup.py
          package.xml
          resource/
                my_ros2_package
          setup.cfg
```

`package.xml` filen må endres. Her må depenencies legges til, slik som her.

```
<?xml version="1.0"?>
<?xml-model href="http://download.ros.org/schema/package_format3.
    xsd" schematypens="http://www.w3.org/2001/XMLSchema"?>
<package format="3">
  <name>lw_data_collector</name>
  <version>0.0.0</version>
  <description>TODO: Package description</description>
  <maintainer email="jorgen@todo.todo">jorgen</maintainer>
  <license>TODO: License declaration</license>

  <depend>rclpy</depend>
  <depend>std_msgs</depend>
  <depend>geometry_msgs</depend>
  <depend>sensor_msgs</depend>
  <depend>numpy</depend>
  <depend>casadi</depend>


  <test_depend>ament_copyright</test_depend>
  <test_depend>ament_flake8</test_depend>
  <test_depend>ament_pep257</test_depend>
```

```
  <test_depend>python3-pytest</test_depend>

  <export>
    <build_type>ament_python</build_type>
  </export>
</package>
```

Når alt dette er fikses kan det MPCen legges til i startup scriptet. Før testing kan det være lurt å kjøre den manuelt fra skapet. SSH for å styre skapet fra ekstern PC. Da er det nødvendig med et modem tilkoblet skapet.

2 Simulator

1. **Åpne terminal**: Start terminalapplikasjonen på ditt Linux-system, helst ved å bruke LoneWolf KDA-datamaskinen (Predator Orion 3000) da filene kan finnes lokalt på denne. Alternativt så er filene lastet opp på Azure under LoneWolf2024

2. **Start MATLAB**: Skriv følgende kommando i terminalen og trykk Enter:

```
    matlab

```

3. **Finn parameterfilen**: I MATLAB, naviger til følgende katalog:

```
    /home/lonewolf/Documents/LoneWolf2024/Simulator/

```

   Finn `LoneWolfparams.m`-filen i denne katalogen.

4. **Åpne og kjør parameterfilen**: Åpne `LoneWolfparams.m`-filen i MATLAB og kjør den.

5. **Åpne Simulink-modell**: I samme katalog, finn `LoneWolfsimulator.slk`-filen. Åpne denne filen i Simulink.

6. **Kjør Simulink-modellen**: I Simulink, trykk på `Run`-knappen for å starte simuleringen.

7. **Kjør Python-fil**: naviger til mappen som inneholder

```
/home/lonewolf/Documents/LoneWolf2024/Simulator/MPC_node_main.
   py}

```

8. **Real time kernel:** Hvis programmet ikke kjører i sanntid, kan dette observeres i et scope nede i venstre hjørne av simulatoren i Simulink, som sammenligner tidssteg med simulatortid. Dette skyldes som regel at real time kernel ikke er installert. Dette gjøres ved å skrive følgende kommando i kommandovinduet i MATLAB:

```
        sldrtkernel -install

```