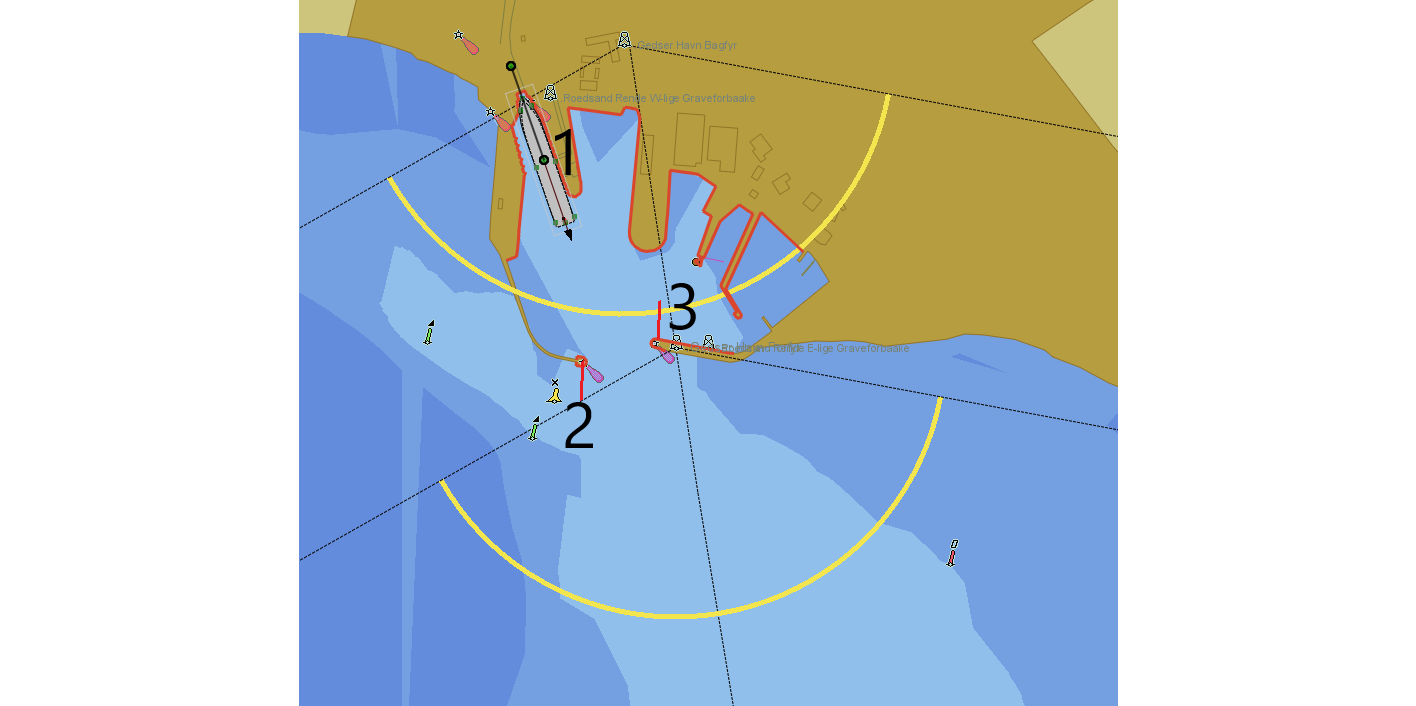
A03.03 Expert tracks

Indledende oplysninger

|  |  |
| --- | --- |
| Indsatsområde | Maritim grøn omstilling og sikkerhed - virtuelle services |
| Institut | FORCE Technology |
| Titel | Objektiv risikovurdering af skibsoperationer |
| Nummerering | A.03.03 |
| Version | Rev 00 |
| Periode | 2021-01-01 - 2021-12-31 |
| Kontaktperson | John Koch Nielsen |

Learning from experts

Intro



Stjæl tekst fra papers…

Most AI algorithms used today to play games uses a reward and tries to optimize this reward, but for many tasks it is difficult to design a reward, just imagine a self driving car, how to reward a overtake situation…

We try here to use VDR recordings form a real ferry to exstract what the captain did.

**Sensor signals available**

|  |  |  |
| --- | --- | --- |
| Column name | Description | Unit |
| Datetime | Date and time |  |
| Lat | Position latitude | Decimal degree |
| N\_S | Letter indicating North/South |  |
| Long | Position longitude | Decimal degree |
| Heading | Heading true (relative North) | Degree |
| COG | Course over ground | Degree |
| SOG | Speed over ground | Knots |
| STW\_UW | Speed through water (longitudinal) | Knots |
| STW\_VW | Speed through water (sidewards) | knots |
| SOG\_U | Speed over ground Longitudinal | knots |
| SOG\_V | Speed over ground transverse | knots |
| Turnrate | Turnrate | Deg/min |
| Depth | Depth below transducer | m |
| W\_dir\_T | Wind direction true | Deg |
| Speed\_MS\_T | Wind speed true | m/s |
| W\_dir\_R | Relative wind direction | Deg |
| W\_Speed\_R\_KN | Relative wind speed | Knots |
| PD\_C | Pitch command |  |
| PD\_S | Pitch statust |  |
| CentRPM\_C | Centre RPM Command | RPM |
| CentRPM\_S\_pct | Centre RPM Status | % |
| CentRPM\_S\_abs | Centre RPM Status | RPM |
| CentRud\_C | Centre rudder command | deg |
| CentRud\_S | Centre rudder status | deg |
| PortM\_Load | Port motor load | ? |
| PortRPM\_C | Port RPM command | RPM |
| PortRPM\_S\_pct | Port RPM status | % |
| PortRPM\_S\_abs | Port RPM status | RPM |
| PortDir\_C | Port direction command | deg |
| PortDir\_S | Port direction status | Deg |
| StbdM\_Load | Starboard motor load | ? |
| StbdRPM\_C | Starboard RPM command | RPM |
| StbdRPM\_S\_pct | Starboard RPM status | % |
| StbdRPM\_S\_abs | Starboard RPM status | RPM |
| StbdDir\_C | Starboard direction command | deg |
| StbdDir\_S | Starboard direction status | Deg |
| Bow1M\_Load | Bow thruster 1 motor load | ? |
| Bow1RPM\_C | Bow thruster 1 RPM command | RPM |
| Bow1RPM\_S\_pct | Bow thruster 1 RPM Status | % |
| Bow2M\_Load | Bow thruster 2 motor load | ? |
| Bow2RPM\_C | Bow thruster 2 RPM command | RPM |
| Bow2RPM\_S\_pct | Bow thruster 2 RPM Status | % |

The absolute position in earth coordinate system is not so relevant as e.g. the position relative to the berth

It is also realized that the captain will position the ship relative to the break water entrance hence a local coordinate system with origo at the berth (1) is introduced and the following values are considered to be the state values used to decide the control actions

**State**

The state of the ship is here extracted to the sensor signals that gives the captain the inputs he needs to decide how to control the ship. Beside direct loggins some additional variables (features) is considered of interest, as e.g. the distance and bearing/direction to the berth as well as to the break waters. Further distance and speed towards/from the leading line(s) may be of importance. In real life the captain clearly sees how well he is lined up for berthing just by looking at the leading lights/poles.

A picture containing letter

Description automatically generated

|  |  |  |
| --- | --- | --- |
| Xe | North coordinate relative to local origo | m |
| Ye | East coordinate relative to local origo | m |
| Dist0 | Distance from ship to origo | m |
| Dir0 | Bearing from ship to origo | Deg |
| Dist1 | Distance from ship origo to point 1 | m |
| Dir1 | Bearing to point 1 seen from ship origo | Deg |
| Dist2 | Distance from ship origo to point 2 | m |
| Dir2 | Bearing to point 2 seen from ship origo | Deg |
| Dist3 | Distance from ship origo to point 3 | m |
| Dir3 | Bearing to point 3 seen from ship origo | Deg |
| DistLead | Distance to leading line (with sign ?) | m |
| Heading | Heading true (relative North) | Degree |
| COG | Course over ground | Degree |
| SOG | Speed over ground | Knots |
| STW\_UW | Speed through water (longitudinal) | Knots |
| STW\_VW | Speed through water (sidewards) | knots |
| SOG\_U | Speed over ground Longitudinal | knots |
| SOG\_V | Speed over ground transverse | knots |
| Turnrate | Turnrate | Deg/min |
| Depth | Depth below transducer | m |
| W\_dir\_T | Wind direction true | Deg |
| Speed\_MS\_T | Wind speed true | m/s |
| W\_dir\_R | Relative wind direction | Deg |
| W\_Speed\_R\_KN | Relative wind speed | Knots |

The position (Xe, Ye) in the local coordinate system is dropped as the same information is found in the polar information distance and direction to origo( dist0, dir0).

File "Z:\Tasks\119\119-25095\Task1.4\Berlin\distancePointsGedserRostock.txt"

#origo=np.array([np.radians(54.57363333), np.radians(11.92475) ]) ## Gedser berth

G1, -88.51367952, 62.97410853

nG2 ,-272.31321847, 154.5955789

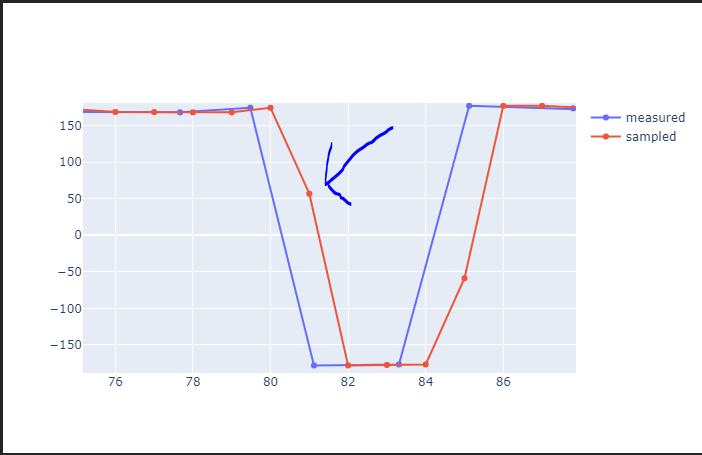
G3 ,-296.4857112 , 73.57014819

Handling of angles

Thr cyclic nature of angles can be difficult for a machine learning system to learn e.g that 1 degree is closer to 359 than 350 is to 359. To handle this all angles are converted to (cos(x) , sin(x))

Sampling of angles

The sampling frequency of the raw signals varies, but for use in the simulator and for machine learning we need signals with constant frequency. To achive this we resample the data with a period of one seconds. If as in the example below there are more than one second between measurements the mean is calculated as to fill in the gap, but for an angle this does not work as en reality the pod moved a few degrees hence we need cos,sin



**Actions**

The captain has different handles to his disposal to control the ship. From the loggings from the real ship we have the actual commanded pitch, propeller revolutions, rudder angle, POD revolutions, POD directions and bow thruster commands, we are not so interested in which handles were used to give the different commands but focus on the actual command values. Here in overview:

|  |  |  |
| --- | --- | --- |
| PD\_C | Pitch command |  |
| CentRPM\_C | Centre RPM Command | RPM |
| CentRud\_C | Centre rudder command | deg |
| PortRPM\_C | Port RPM command | RPM |
| PortDir\_C | Port direction command | deg |
| StbdRPM\_C | Starboard RPM command | RPM |
| StbdDir\_C | Starboard direction command | deg |
| Bow1RPM\_C | Bow thruster 1 RPM command | RPM |
| Bow2RPM\_C | Bow thruster 2 RPM command | RPM |

Extracting states and actions

From VDR recordings

Sensor signals are stored in a number of NMEA log files, where all sensors NMEA strings are stored with a time stamp.

In the ShippingLab project a script was developed to extract each sensor en separate files, then identifying when the ship was sailing to filter all recordings for laying at berth out.

The sensors are clearly logged at different sampling intervals and when defining a state it is needed to time align the values. ShippingLab developed a script for that, and here a RK2021 variant is written to extract states and actions as defined above.

**Astern out of port**

First attempt is to save save states until the bearing towards the easterly pier (Point 2) is 90 deg

Script used is

"Z:\Tasks\121\121-20658\04 Technical\extractExpertTracksFromVDR.py"

Data saved here:

"Z:\Tasks\121\121-20658\04 Technical\GedserExpertTracks3points" contains distances to piers and berth

"Z:\Tasks\121\121-20658\04 Technical\GedserExperTracksLeadingLine" contains distance to leading line

"Z:\Tasks\121\121-20658\04 Technical\GedserExpertTracksWithTurn" contains distance to leading line

Ends after turn

Training of NN

Følgende script er lavet til at træne et standard feed foward NN

H:\GitRepos\PyDriver\tests\btjNN.py

Using trained Neural net

The script

H:\GitRepos\PyDriver\tests\runTrainedAInetwork\_dk1driver.py

Is used to run simflex with a pretrained neural net (using pytorch )

Plotting results

**Analyser**

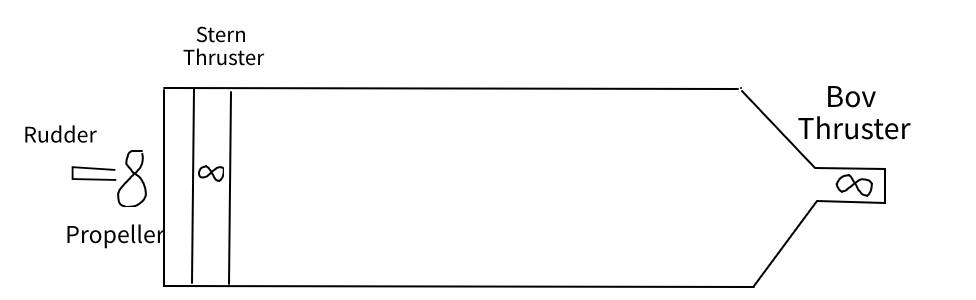
Typically Ship types and propulsion systems

To give a rough overview of typical ship types and how they are configured some sketch’es below (sorry for the quality, but then no owner rights are violated)

**Cargo/tanker/container**

These ship types are typically have one propeller and one rudder.

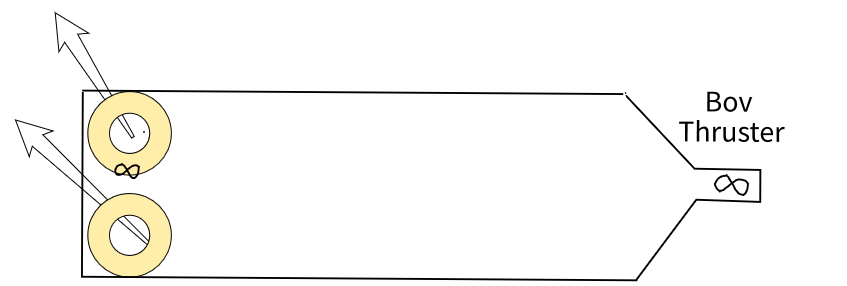
They can have bow and stern thruster but differs with size of ship



**Pod-ships -> Tugs, Ferrys, cruise liners, special offshore**

One or more azimuthing propellers able to give force in any direction and eventually bow thrusters

Older Ferrys and cruise ships often have rudder propellers instead of “POD”

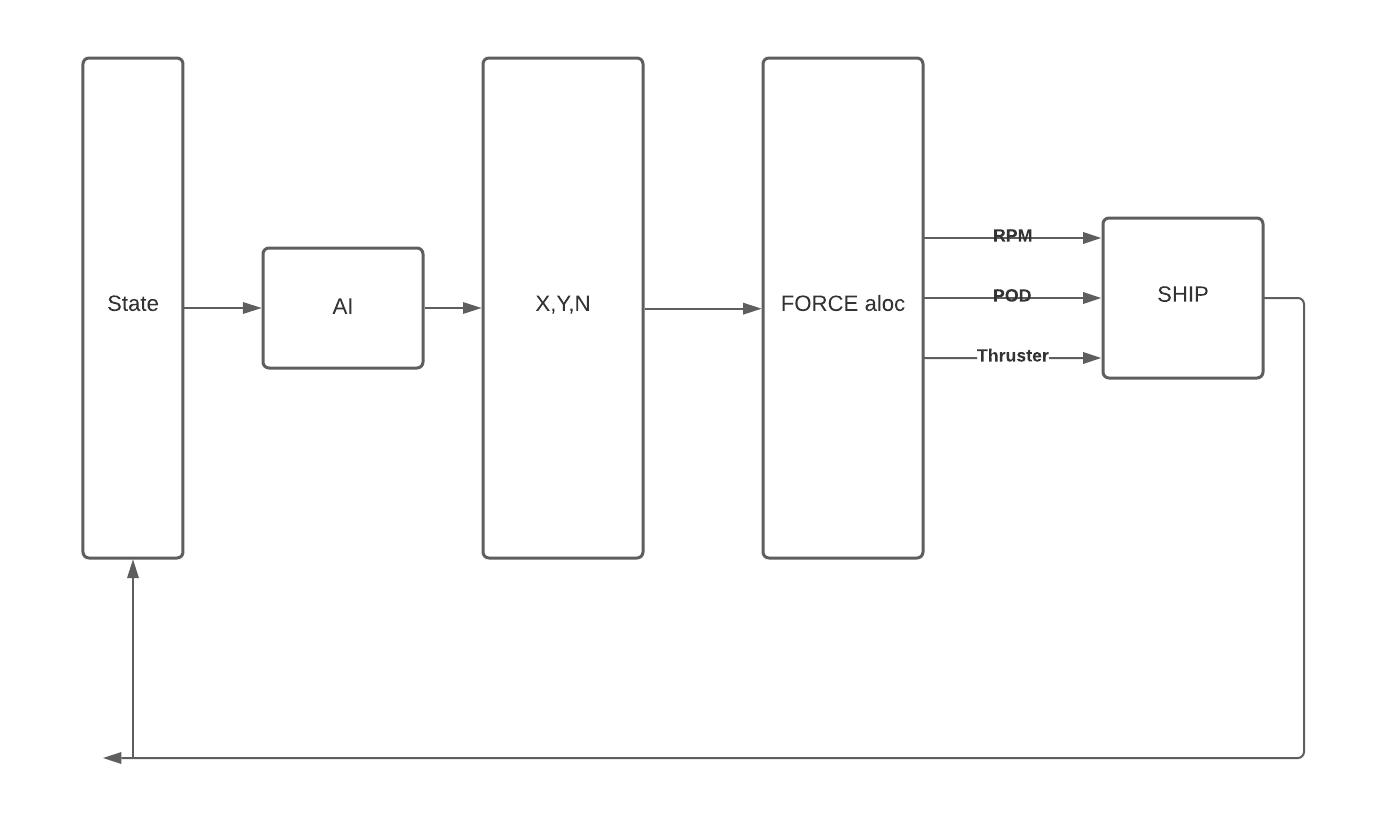


Generalisation

During our discusion FT proposed to generalize the aoutpout from the AI to be forces (Force X,Y and Moment N).

Then no matter how the ship is equipped with force giving units FT can develop a force allocater that utilize the equipment on the actual ship to generate the requested forces. That part is this way reusable for any ship even tugs can be allocated to assist in realizing the requested force.

Diagram sketch below. The state (maybe even the history) as input to the AI. Output forces feed onto the force allocator that converts the force request to rudder angles rpms etc



From measurements of RPM, rudder angles etc FORCE can calculate the resulting forces that can be used to train the network.

**Generalisation of the state**

Generalisationof the “state” is not solved yet and here follow another example of a port investigation

**Case Port of Rønne**

“Is there sufficient space for a wind mill installation ship to operate in Port of Rønne”

The installation ship has to berth at the SE berth in the new port of Rønne. The ship follows the leading line towards the berth. About one ship length before the berth the ship turns 90 degrees and moves sidewarths (using thrusters and POD’s) towards the berth.

When depaturing the ship moes sidewarts and when at a sufficient distance it turns 90 degree on the spot. Position itself at the leadingline and slowly leaves the port.

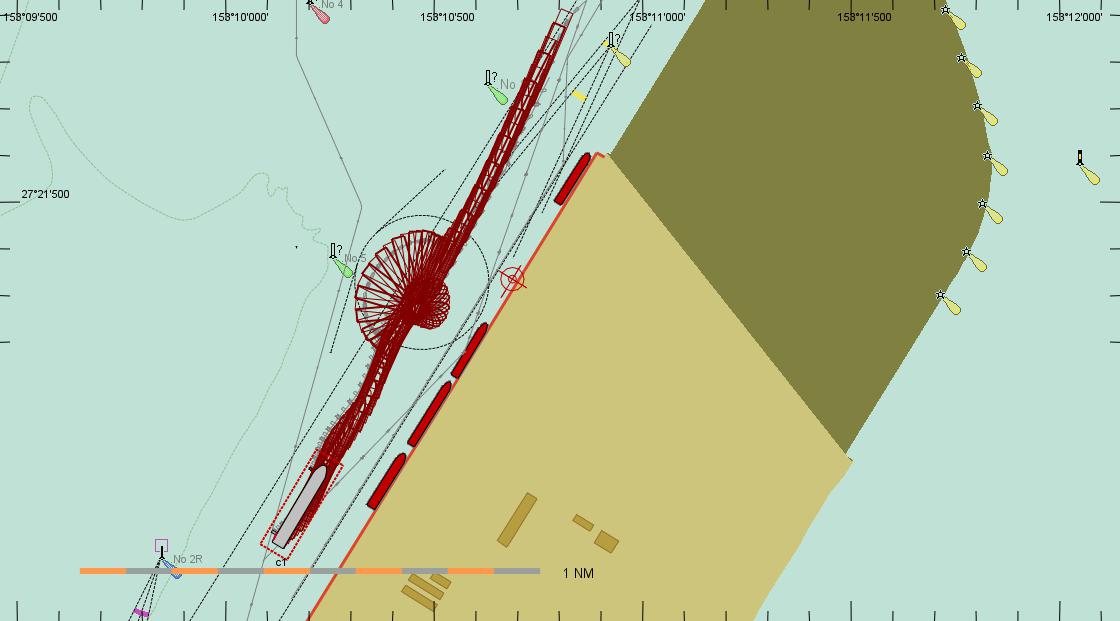


**Case Brisbane**

**“**Sufficient size of turning basin”

Container ships arrive from North assisted by two tugs. When inside the dedicated turning basin the ship is turned 180 degrees then moved astern until parallel with the dedicated berth and lasly pushed sidewarts to the berth by the tugs

Here bouys and the “map” indicate where the ship can sail, not sure about leadinglines, but turning area is marked with the bouy and the map



**Case Vordingborg**

**“**Can the expected size of ships call the port safely”

Bulk carries call Vordingborg assisted by two tugs. When outside the port entrance the ship is turned by the tugs and pushed into the final berth position





Papers and background material

Inspiration videos

!!!!START HERE!!!! : Very good video explaining IRL

Safe and efficient Inverse Reinforcement learning

https://www.youtube.com/watch?v=Ob\_tcEzVyjU

-------------------------------------------------

Breakthrough experiment with model helicopter

**Autonomous Helicopters Teach Themselves to Fly Stunts**

<https://www.youtube.com/watch?v=M-QUkgk3HyE>

acm09-ApprenticeshipLearningHelicopterControl.pdf

Papers etc attached

**Robert learning from a human expert using inverse reinforcement learning**

IRL\_VT4.pdf

Code repository excist but a bit messy, I failed to figure out what to use if any

**Inverse Reinforcement learning**

bootcamp\_inverserl.pdf

*Slide show with some overview*

**Maximum Entropy Inverse ReinforcementLearning**

MaximumEntropyInverseReinforcementLearning.pdf

*The paper deals with a discrete universe. Can the algoritm be used for continuous states and actions ?*

*No code examples found*

**Guided cost learning**

GuidedCostLearning.pdf

*Looks like this could be used, but no code examples found. Algo’s given*

**Generative adversarial imitation learning**

NIPS-2016-generative-adversarial-imitation-learning-Paper.pdf

*Found some gail.py implementations on github and started out with that but ended with simple training of a NN ?!? Is GAIL the way to proceed ??*

**Autonomous ship Navigation Methods a Review**

Autonomous\_Ship\_Navigation\_Methods\_A\_Review.pdf

*List different methods used. Our project is more concerned about low speed manoeuvering…*

**Learning robust rewards**

LearningRobustRewards.pdf

*Adveserial Inverse reinforcement learning AIRL sounds very promising to our problem but no working implementation code found yet.*

**Learning from Demonstrations**

*LearningFromDemonstrations.pdf*

*Looks promising but no code examples found*

**Inverse reinforcement learning in continuous time and space**

InverseReinforcementlearninInContinuousTimeAndSpace.pdf

*Yes we want that algo. Pseodo code given, no implementation found*

**Deep Multi Intensions Inverse Reinforcement Learning**

DeepMultipleIntensionsInverseReinforcementLearning.pdf

*Seems only used for discrete envirs*

**Machine learning method selection for velocity error estimation on a ferry**

ShippingLab D1.4.4-A-MachineLearning.pdf

*Deals with difference between simulator model and real ship. Machine learning method used to estimate speed difference*