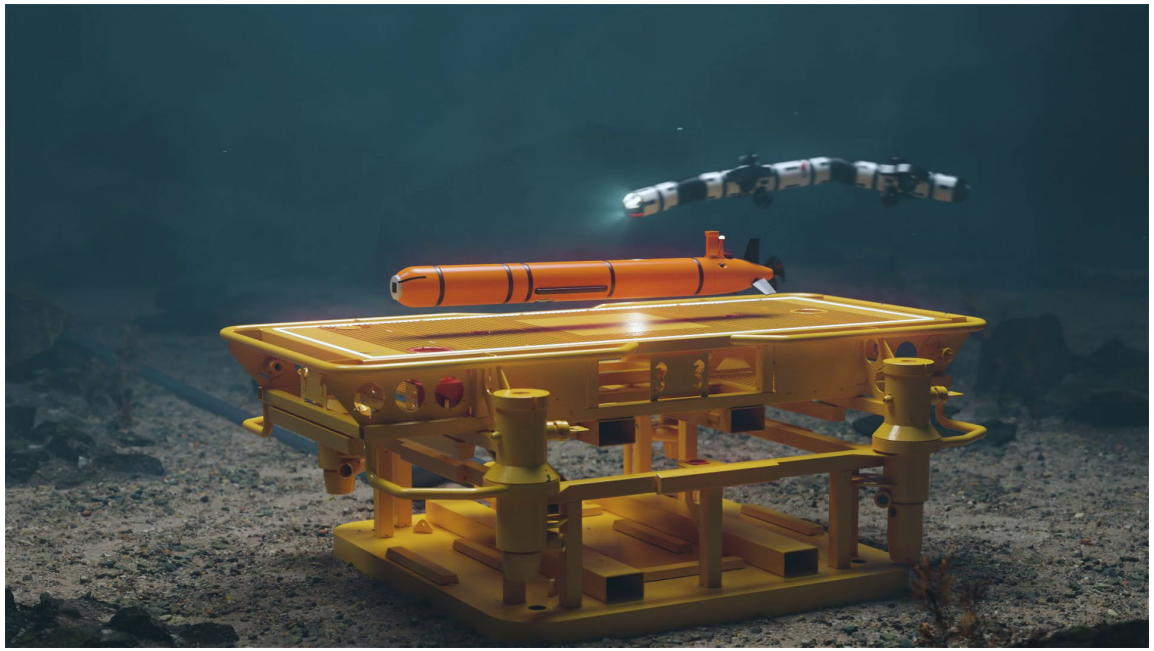
TMR4585

Autonomy 2019

Martin Ludvigsen
Applied Underwater Robotics Laboratory (AUR-Lab)
Department of Marine Technology
Norwegian University of Science and Technology
E-mail: martinl@ntnu.no

NTNU

Kahoot



Learning objectives autonomy

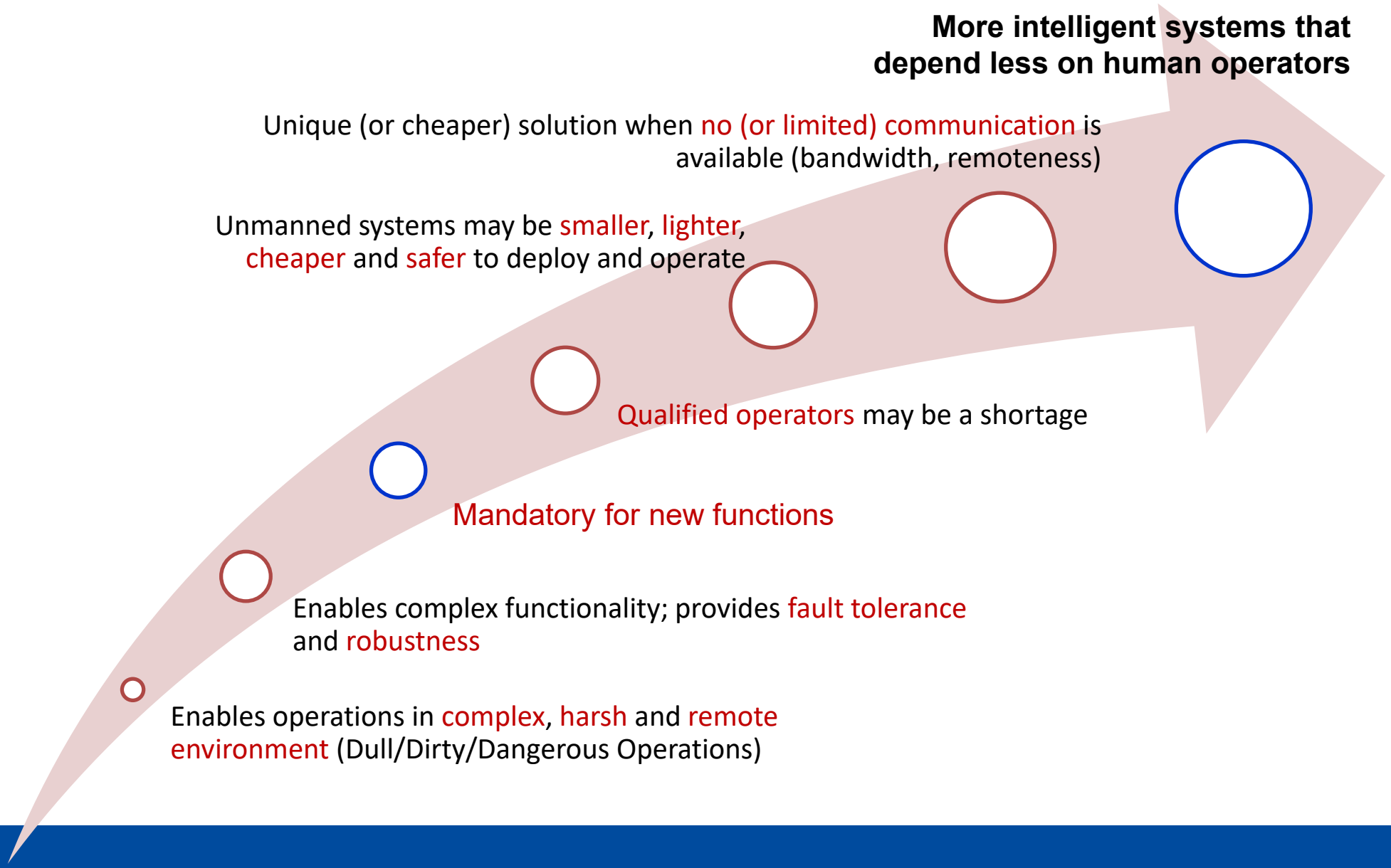
- Control architecture for underwater vehicles
- Levels of autonomy
- Architectures
  - Reactive and deliberative systems
- Behaviors
- Research examples

Areas of autonomy

1. Energy
2. Navigation
3. **Decisions**

- Motivation for high level autonomy
- Improved endurance
- Complex dynamic environment
- Complex missions
- UUV specific missions - under ice

Why autonomy?

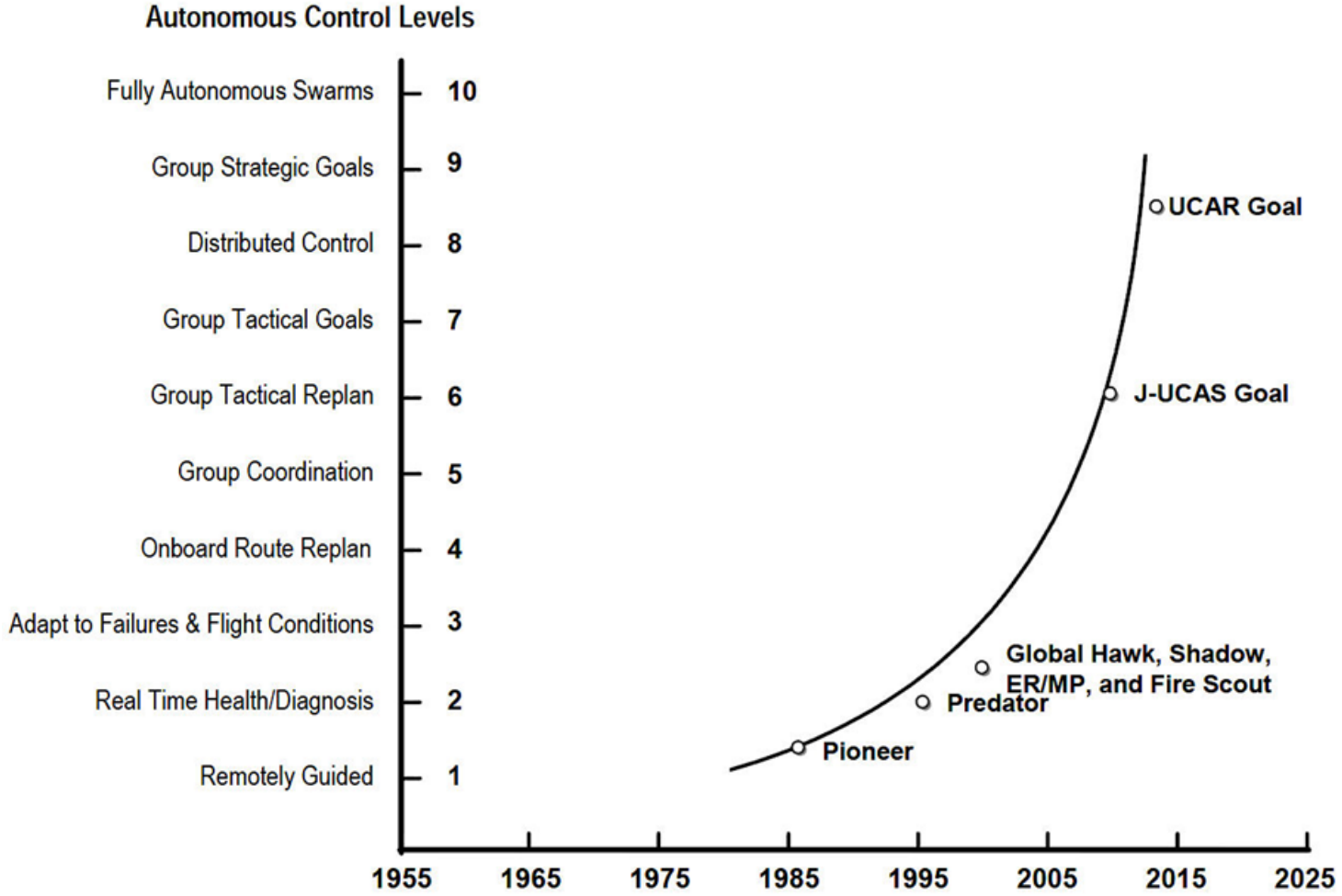


Autonomy space

(Seto et al 2013)

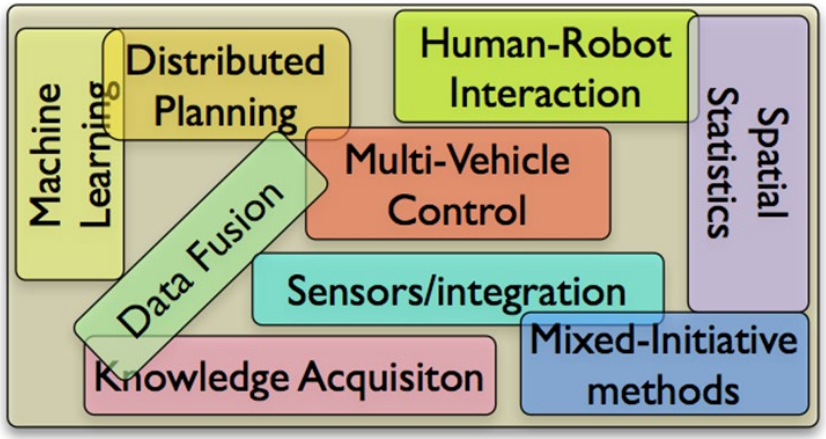
- Situation awareness
  - Raw
  - Semi-processed
  - Feature
  - Aggregate
  - Interpreted
  - Inferred
  - Intent
- Decision making, planning and control
  - Direct control
  - Sequenced control
  - Adaptive execution
  - Objective achievement
  - Multiple objective
  - Joint objective achievement
- External interaction
  - Teleoperations
  - Remote control
  - Semi-autonomous
  - Fully autonomous

Another definition with 10 levels ..



Bottom-up vs. top-down approach to autonomy

- In order to develop autonomous robots, the focus of engineering cybernetics must be widened to include methods for cognitive planning, typically through methods within artificial intelligence (AI)
- Considerations by autonomy expert Kanna Rajan:
  - "While AI has often been in the public imagination and associated with robotic platforms, the field's impact on real world problems especially in robotics has, till recently, been relegated to interesting laboratory methods that do not scale to real world environments. This is changing, with top-down abstractions in AI meeting bottom-up methods coming from Robotics."
  - "While AI is a conglomeration of techniques, the most relevant towards the key goal of ocean exploration, observation and monitoring are, we believe, Automated Planning and Execution, Machine Learning, Autonomous Agents and Diagnosis."



Levels of autonomy as defined by the Uninhabited Combat Air Vehicle Program

Level 1 (Manual Operation)

- The human operator directs and controls all mission functions.
- The vehicle still flies autonomously.

Level 2 (Management by Consent)

- The system automatically recommends actions for selected functions.
- The system prompts the operator at key points for information or decisions.
- Many of today's autonomous vehicles operate at this level.

Level 3 (Management by Exception)

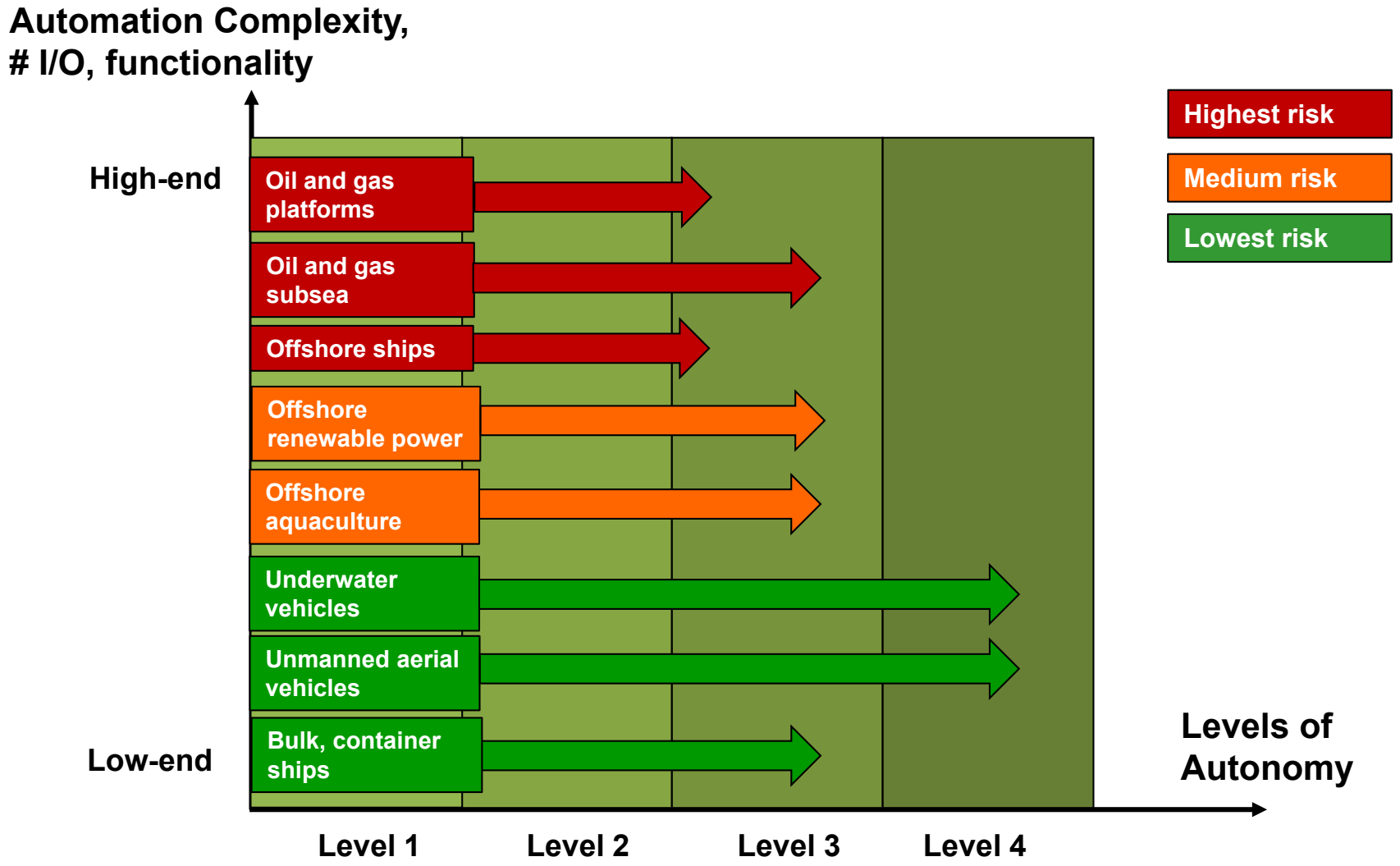
- The system automatically executes mission-related functions when response times are too short for operator intervention.
- The operator is alerted to function progress.
- The operator may override or alter parameters and cancel or redirect actions within defined time lines.
- Exceptions are brought to the operator's attention for decisions.

Level 4 (Fully Autonomous)

- The system automatically executes mission-related functions when response times are too short for operator intervention.
- The operator is alerted to function progress.

Autonomous Vehicles in Support of Naval Operations, http://www.nap.edu/catalog/11379.html

Candidates for autonomy.....

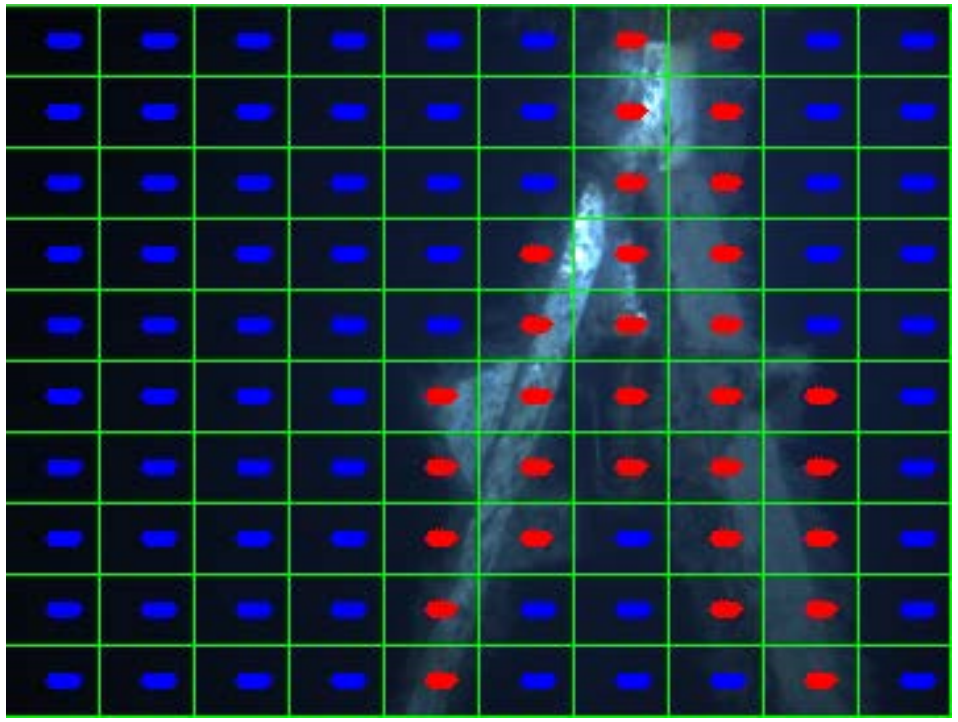


How to achieve Autonomous Marine Operations and Systems? Disciplines and Methods

- **Mathematical modelling** based on hydrodynamics, aerodynamics, structural mechanics and electro-mechanical systems will be achieved through a systems perspective integrating models and knowledge from the different domains.
- **Real-time estimation of states and parameters** in order to adaptively update models to detect normal and abnormal changes in the systems or their environment.
- **Advanced sensor fusion for perception** of the environment and any object of interest (OOI) will include integration of imaging sensors such as radar, optics, and acoustics with inertial and navigation sensors for accurate detection and tracking of objects and environmental parameters.
- Model-based optimization will be realized with **coordinated control and robust networked communication** in complex environments with simultaneous operations, robotics, and mobile sensor networks.
- **Integrated guidance and planning** with high-level mission planning will be achieved using numerical optimization where data, decisions, rules and models are represented as constraints, as well as tools from artificial intelligence such as discrete search algorithms.
- Intelligent control command and task execution with **fault-detection and diagnosis** as a basis for **reconfigurable control** and re-planning.

Arcitectures

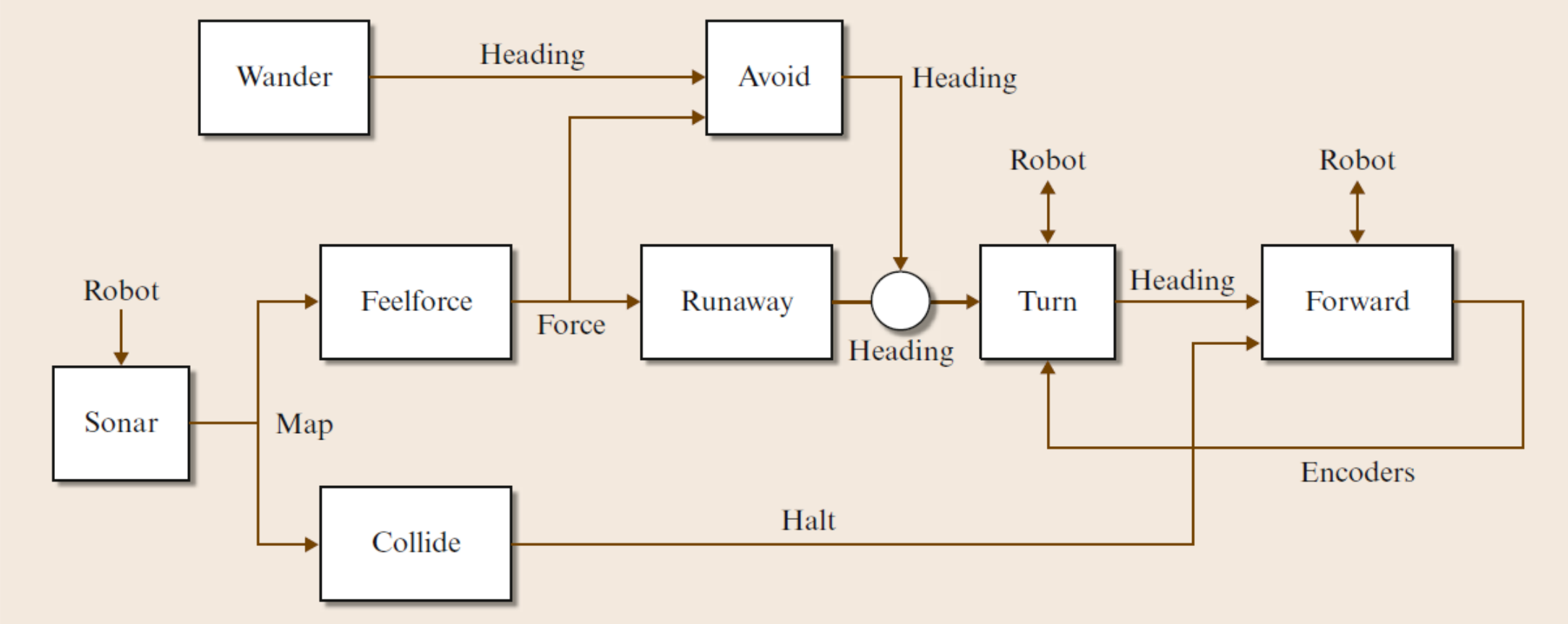
- Subsumption
- State machine
- Layered robot control



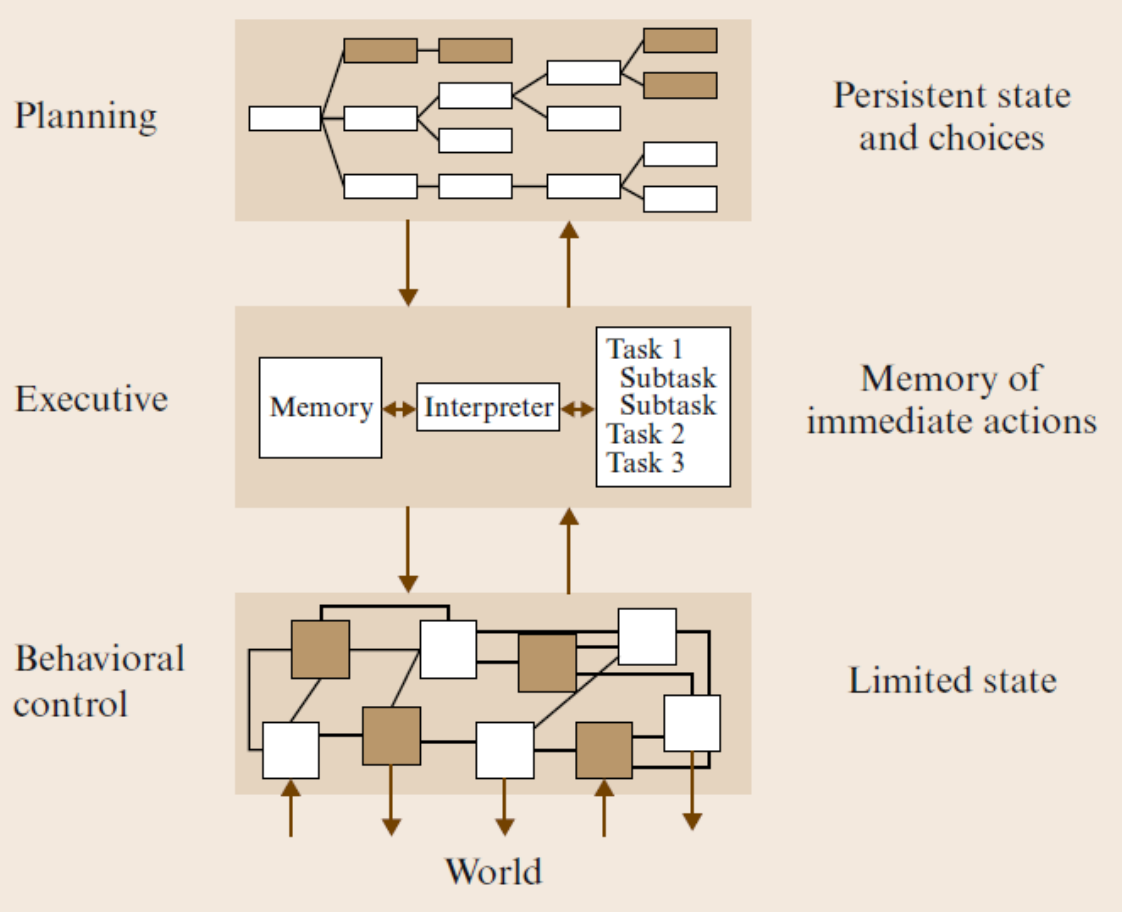
Subsumption

- Subsumption had an *arbitration* mechanism that enabled higher-level behaviors to override signals from lower-level behaviors.
- The robot might have a behavior that drives the robot in random directions.
- This behavior is always active and the robot is always driving somewhere.
- A second, higher-level behavior could take sensor input, detect obstacles, and steer the robot away from them. It is also always active.
- In an environment with no obstacles, the higher-level behavior never generates a signal.
- if it detects an obstacle it overrides the lower-level behavior and steers the robot away.
- As soon as the obstacle is gone (and the higher-level behavior stops sending signals), the lower level behavior gets control again.
- Multiple, interacting layers of behaviors could be built to produce more and more complex robots.

Subsumption



Layered arcitecture



- Planning
- Sequencing
- Real-time control
- The executive layer coordinates the behavioral layer to achieve tasks
- The task-planning layer is responsible for deciding the order of deliveries to minimize time. The task-planning layer sends tasks to the executive.

Behavioral Control

- Behavioral control represents the lowest level of control in a robot architecture.
- In architectures such as Subsumption, all behaviors are running concurrently with a hierarchical control scheme inhibiting the outputs of certain behaviors.

- Connecting points
- Behavoiral control
- Executive
  - Interface between the numerical behavioral control and the symbolic planning layers.
  - Responsible for translating high-level plans into low-level behaviors, invoking behaviors at the appropriate times, monitoring execution, and handling exceptions.
  - Allocate and monitor resource usage, although that functionality is more commonly performed by the planning layer.
- Planning
  - Responsible for determining the long-range activities of the robot based on high-level goals.

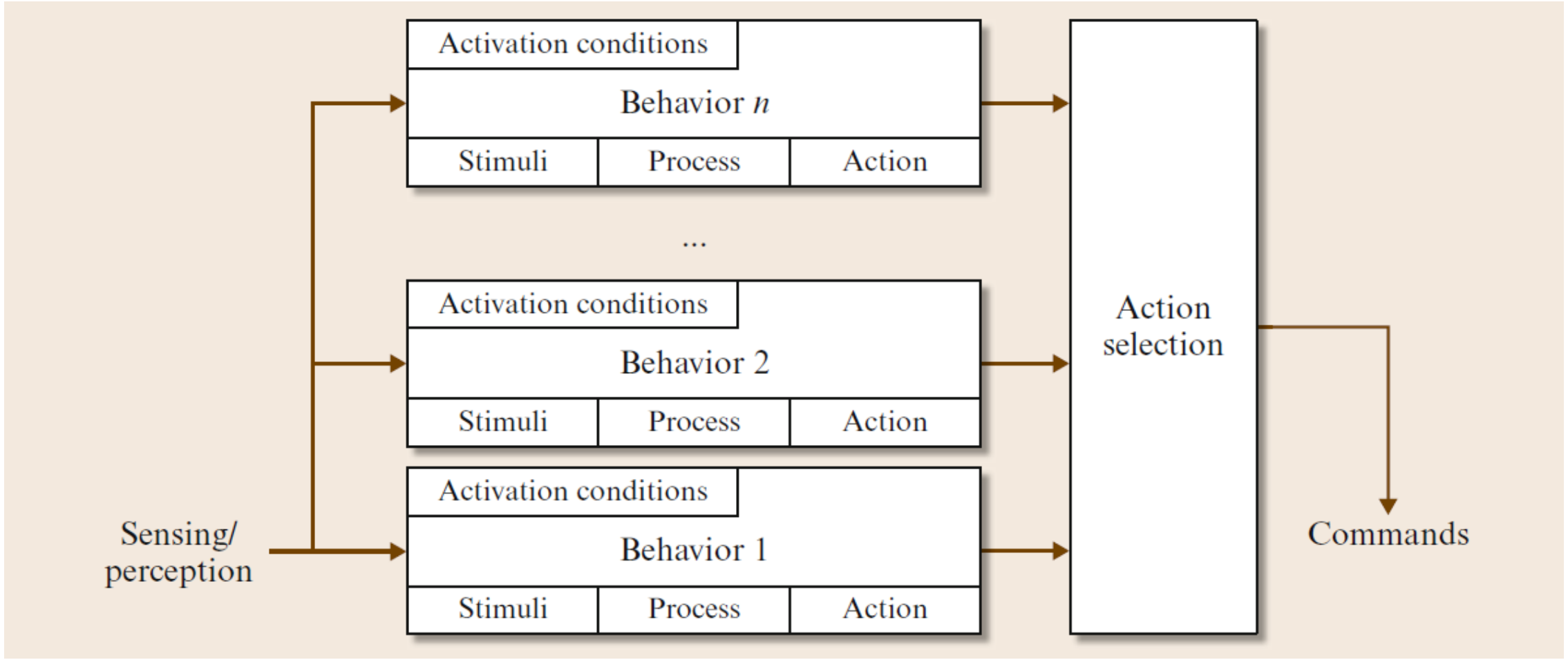
Behaviour based control

- Deliberative
- Reactive
- Hybid

Basic principles of behavior-based control

- Behaviors are implemented as control laws (sometimes similar to those used in control theory), either in software or hardware, as a processing element or as a procedure
- Each behavior can take inputs from the robot's sensors (e.g., proximity sensors, range detectors, contact sensors, camera) and/or from other modules in the system, and send outputs to the robot's effectors (e.g., wheels, grippers, arm, speech) and/or to other modules
- Many different behaviors may independently receive input from the same sensors and output action commands to the same actuators.
- Behaviors are encoded to be relatively simple, and are added to the system incrementally.
- Behaviors (or subsets thereof) are executed concurrently, not sequentially, in order to exploit parallelism and speed of computation, as well as the interaction dynamics among behaviors and between behaviors and the environment.

Schematic of behavior-based approach



A Finite State Machine

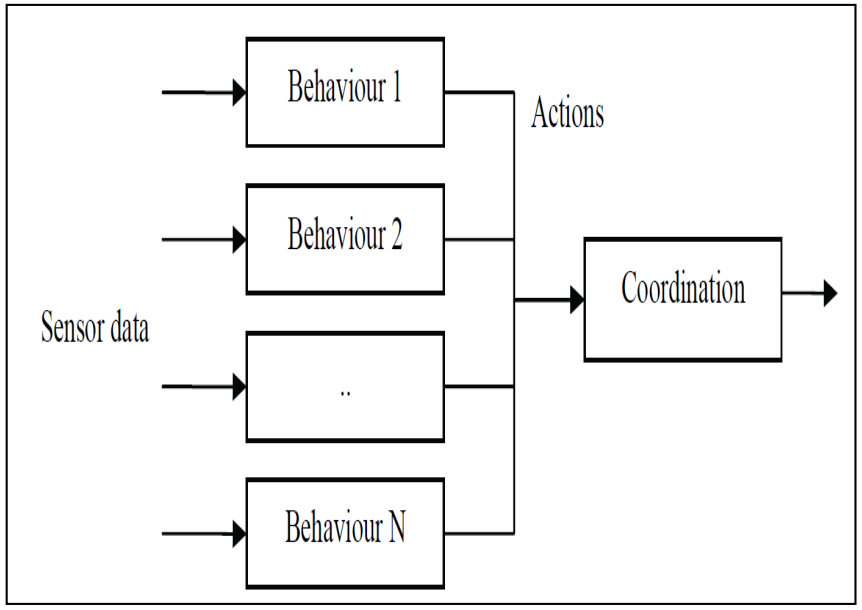
- A reactive system whose response to a particular stimulus (a signal, or a piece of input) is not the same on every occasion, depending on its current “state”.
- For example, in the case of a parking ticket machine, it will not print a ticket when you press the button unless you have already inserted some money.
- Thus the response to the print button depends on the previous history of the use of the system.

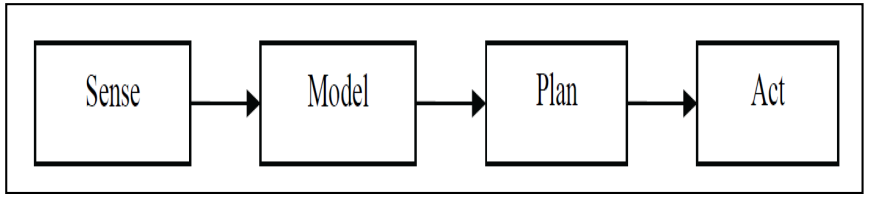
Implement a state machine

- The Finite State Machine class keeps track of the current state, and the list of valid state transitions.
- You define each transition by specifying :
  - FromState - the starting state for this transition
  - ToState - the end state for this transition
  - condition - a callable which when it returns True means this transition is valid
  - callback - an optional callable function which is invoked when this transition is executed.

Autonomy architectures

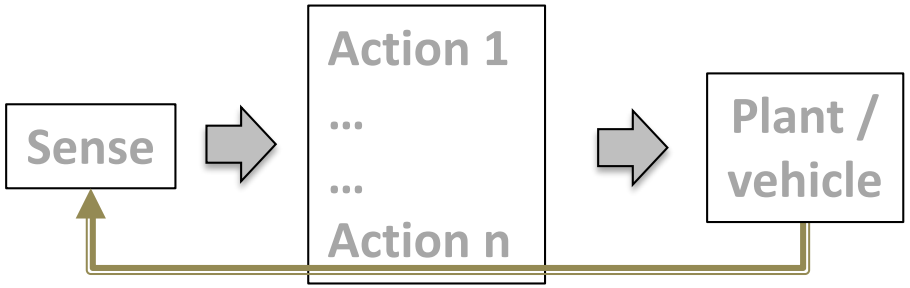
- Reactive control layer
  - Robustness
- Deliberate control layer
  - Optimization
- Hybrid system





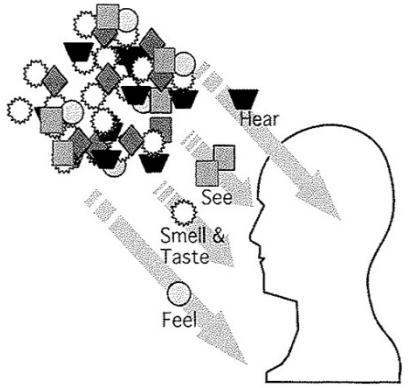
Reactive control systems

- Current systems are very good at solving precisely defined problems, where carefully designed algorithms precisely define the relationship between measurements ("sense") and actions 1,…,n ("act")
- Current systems are reactive, and have preprogrammed a direct relationships of the type "sense => act"
- Examples of reactive systems include:
  - Autopilots, LOS
  - Dynamic positioning (DP) systems
  - Contingency handling: anti collision, alarms systems, …

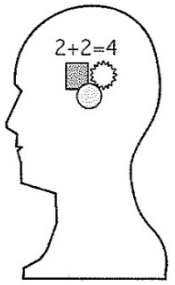


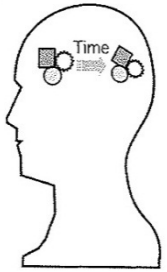
Situation awareness

- Being aware of what is happening around you and understanding what this information means to you now and in the future
- The formal definition breaks down into three separate levels:
  - Level 1: **Perception** of the elements in the environment
  - Level 2: **Comprehension** of the current situation
  - Level 3: **Projection** of the future situation
- To be implemented in appropriate system models



Designing for Situation Awareness. An Approach to User-Centered Design. Endsley, Bolte, Jones



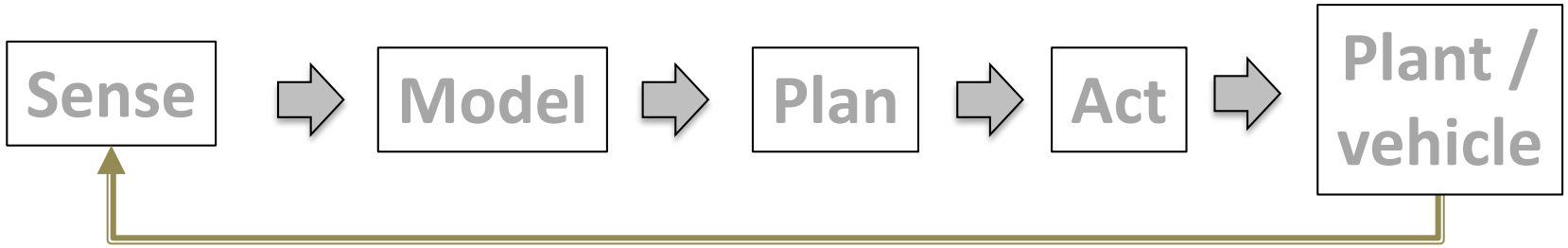


Deliberative control architecture

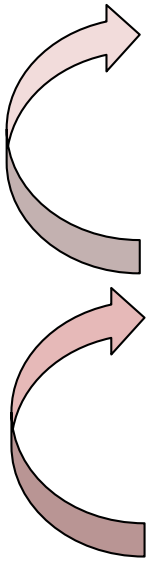
Going from reactive to proactive systems of the type:
“Sense => Model => Plan => Act"

Learning by model updates and accumulation of knowledge

- Situation awareness
- Learning by sensing and observing
- Learning by doing
- Human-in-the-loop



Combining reactive and deliberative control



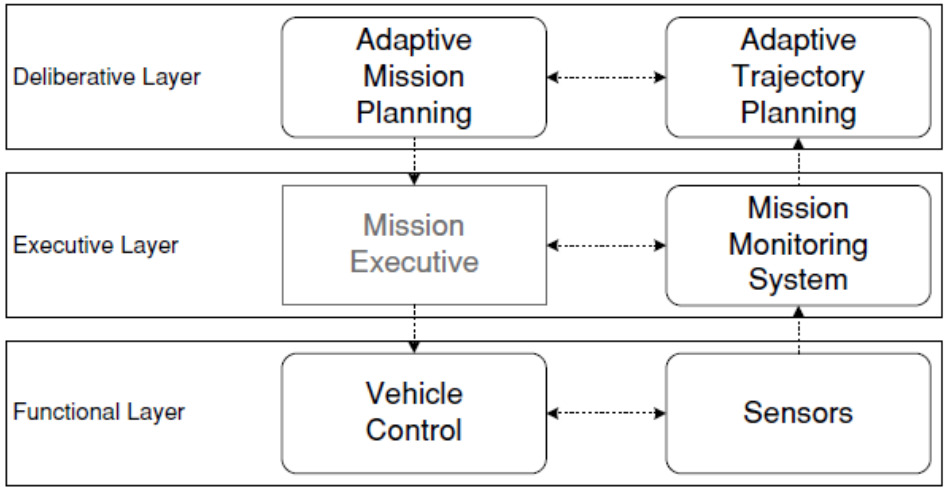
Mission planner level: Mission objective is defined and the mission is planned. Subject to contingency handling, any input from payload sensor data analysis and any other input from the autonomy layer, the mission may be re-planned.

Guidance and optimization *level* handles waypoints and references commands to the controller.

Control execution level: at this level the plant control and actuator control takes place

AUV control systems

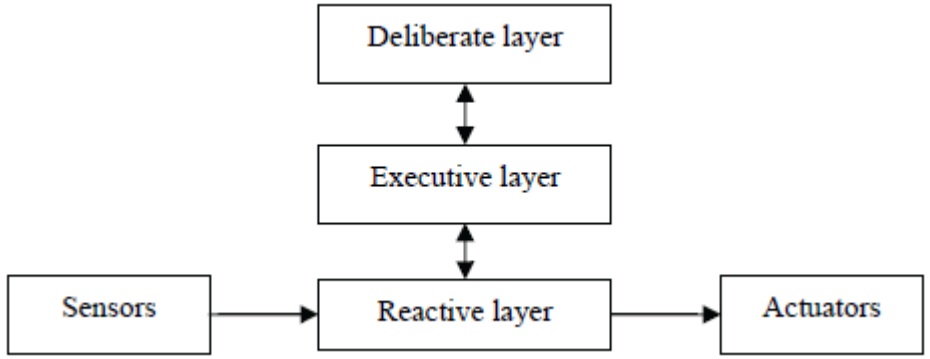
- Mission management
- Sequential scripts
  - Waypoints
  - Track lines
  - Objectives
- Controllers for heading, speed, roll, depth



Architectures

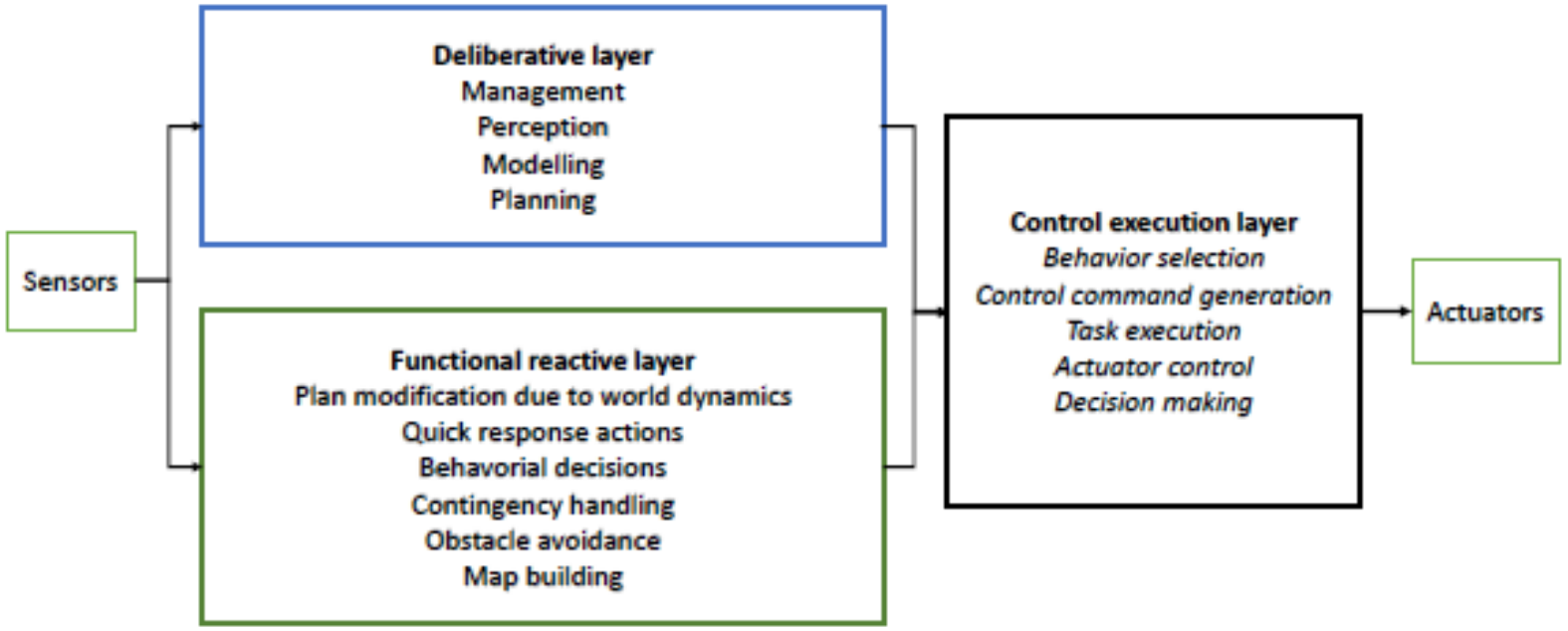
Hybrid models

- Reactive low level
- Deliberative higher levels

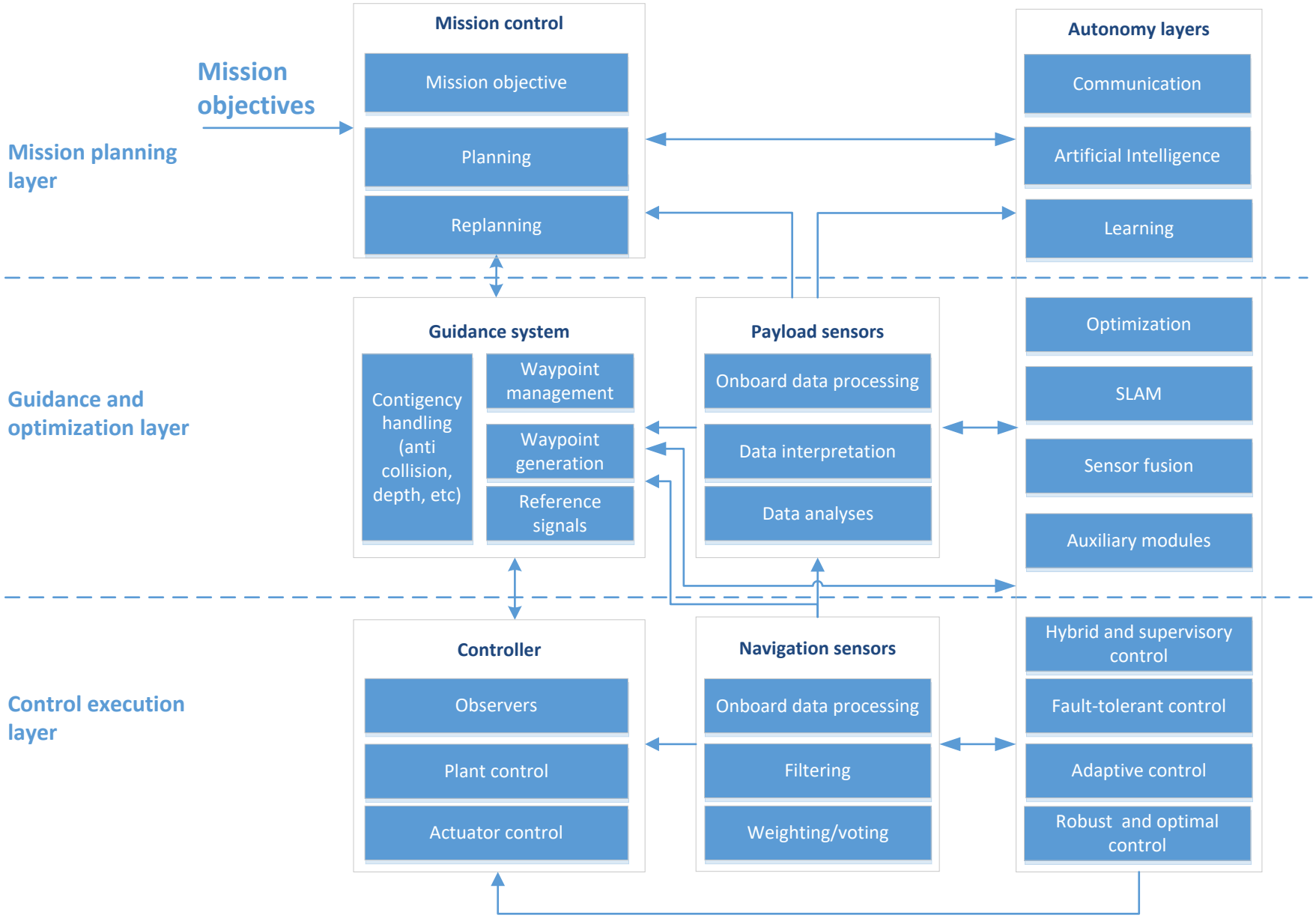


Hagen et al. 2008

Hybrid control layer



Control architecture for underwater vehicles



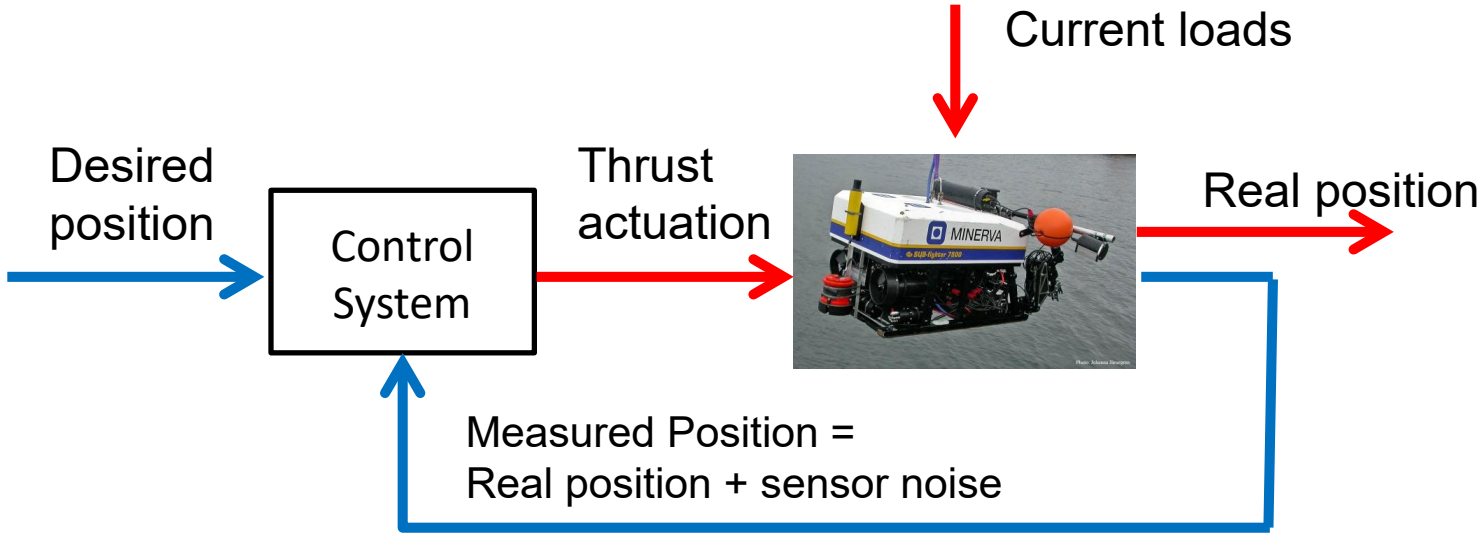
Choice of Robot Architectures

- What are the tasks the robot will be performing?
  - Are they long-term tasks?
  - Short-term?
  - User-initiated?
  - Robot-initiated?
  - Are the tasks repetitive or different across time?
- What actions are necessary to perform the tasks?
  - How are those actions represented?
  - How are those actions coordinated?
  - How fast do actions need to be selected/changed?
  - At what speed do each of the actions need to run in order to keep the robot safe?
- What data is necessary to do the tasks?
  - How will the robot obtain that data from the environment or from the users?
  - What sensors will produce the data?
  - What representations will be used for the data?
  - What processes will abstract the sensory data into representations internal to the architecture?
  - How often does the data need to be updated? How often can it be updated?
- What computational capabilities will the robot have?
  - What data will these computational capabilities produce?
  - What data will they consume?
  - How will the computational capabilities of a robot be divided, structured, and interconnected?
  - What is the best decomposition/granularity of computational capabilities?
  - How much does each computational capability have to know about the other capabilities?
  - Are there legacy computational capabilities (from other robots, other robot projects, etc.) that will be used?
- Who are the robot's users?
  - What will they command the robot to do?
  - What information will they want to see from the robot?
  - What understanding do they need of the robot's computational capabilities?
  - How will the user know what the robot is doing?
  - Is the user interaction peer to peer, supervisory, or as a bystander?
- How will the robot be evaluated?
  - What are the success criteria?
  - What are the failure modes?
  - What is the mitigation for those failure modes?
- Will the robot architecture be used for more than one set of tasks?
  - For more than one kind of robot?
  - By more than one team of developers?

Robot architectures

- Designed to facilitate the concurrent execution of task-achieving behaviors.
- Enable systems to control actuators, interpret sensors, plan, monitor execution, and deal with unexpected contingencies and opportunities.
- Provide the conceptual framework within which domain-dependent software development can take place, and they often provide programming tools that facilitate that development.

Feedback control



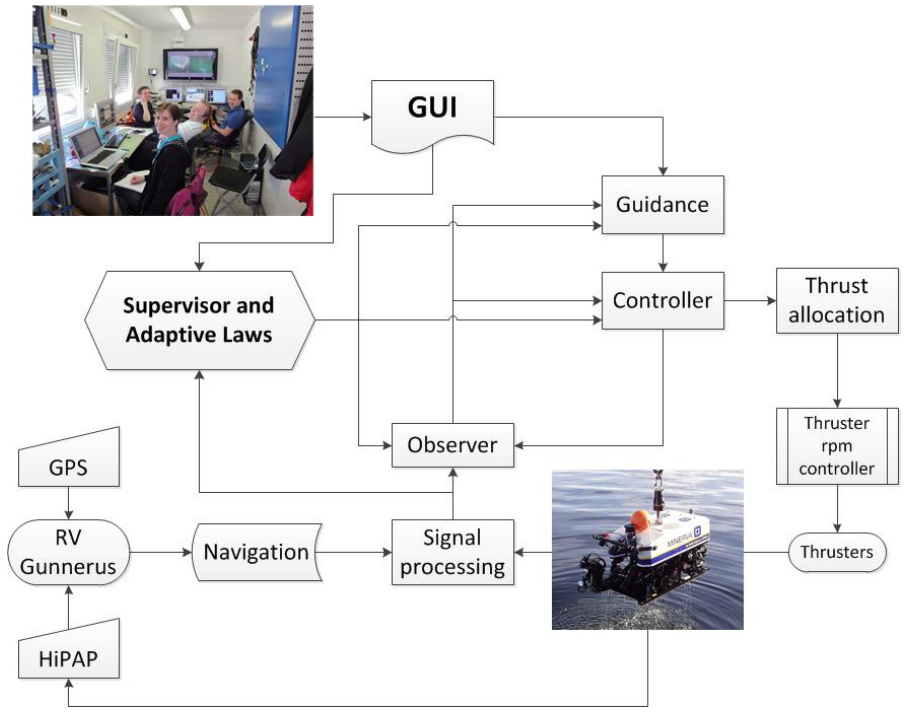
The control system will dominate the dynamics, and we have to consider a new dynamic system – "closed-loop system"

The control system is also a dynamic system with

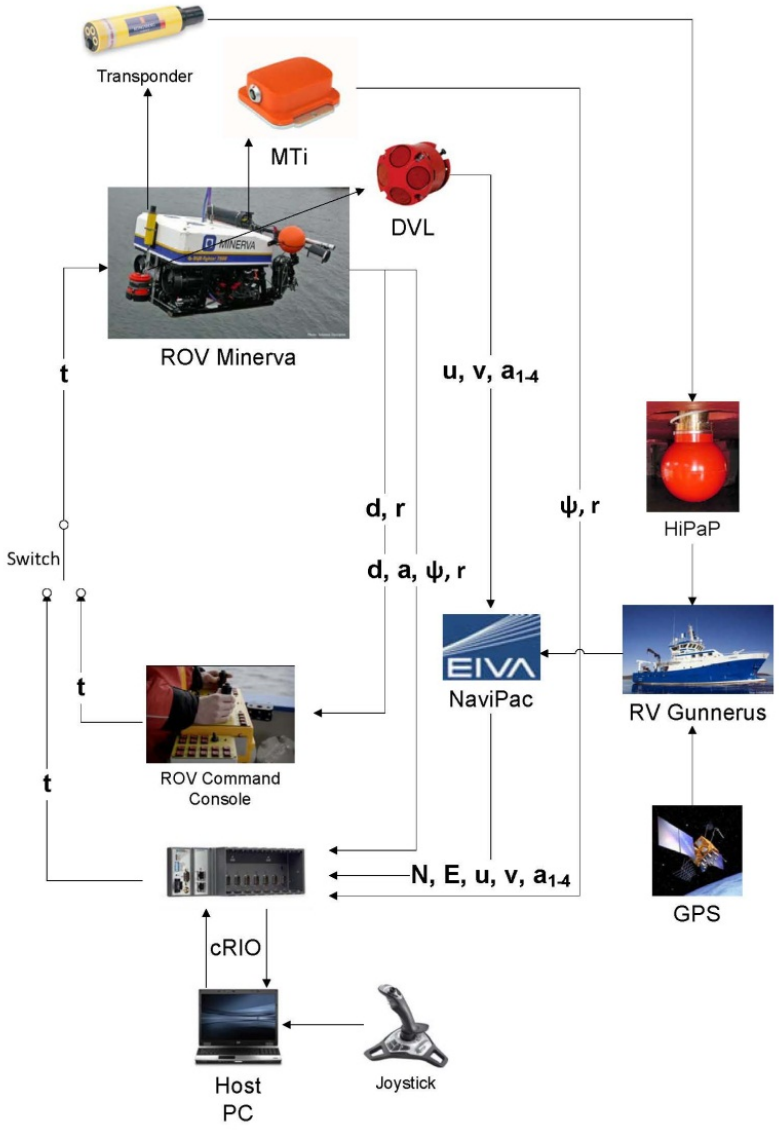
- internal states – state feedback or output feedback using observers
- actuator dynamics, saturation, control allocation, power limitation
- sensor dynamics, integration and accuracy
- computers conducting real time control

ROV control system

- Reference signal
- Measurements
- Actuators
- Process
- Feedback

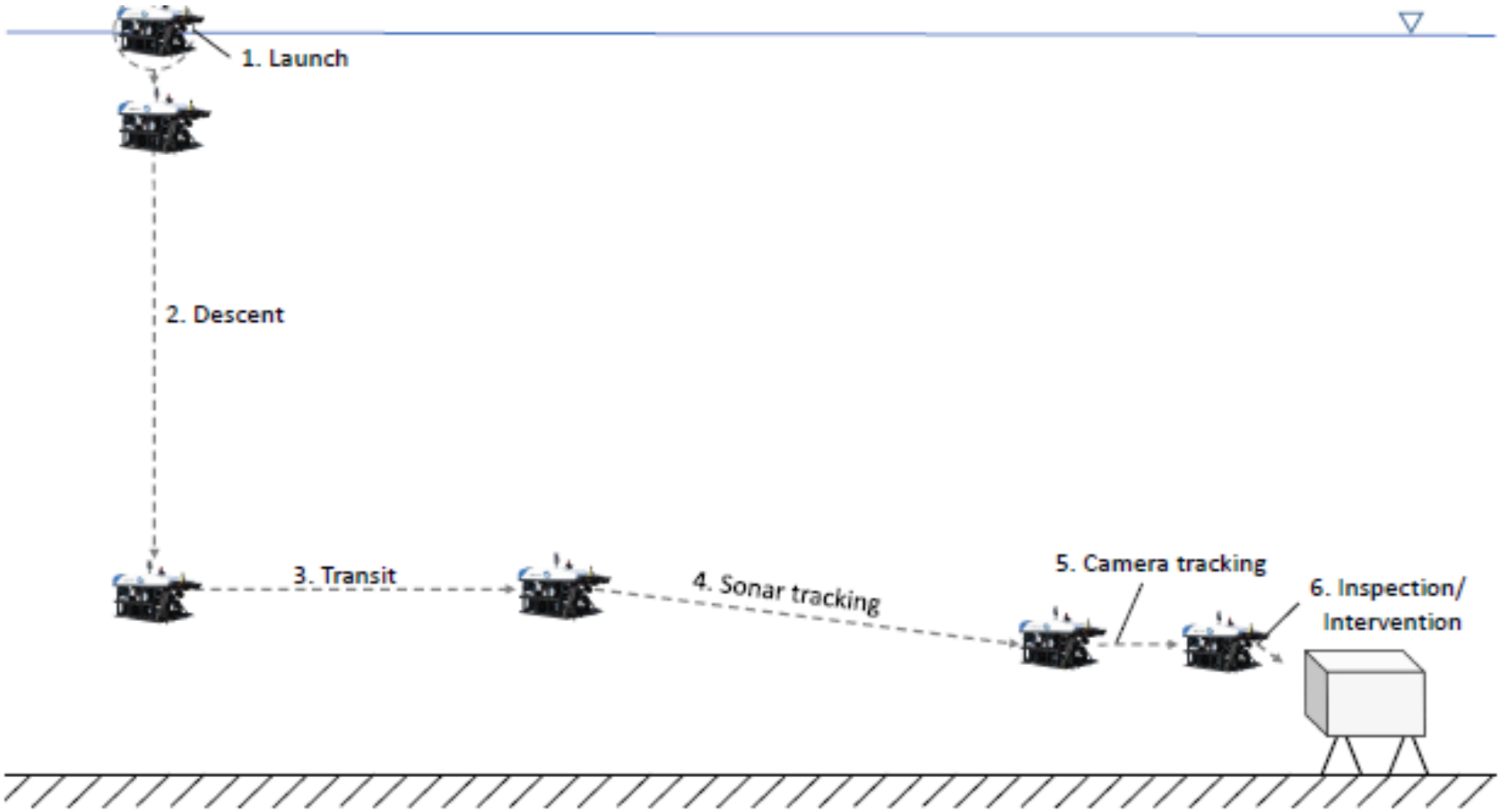


ROV Hardware and Signal Flow

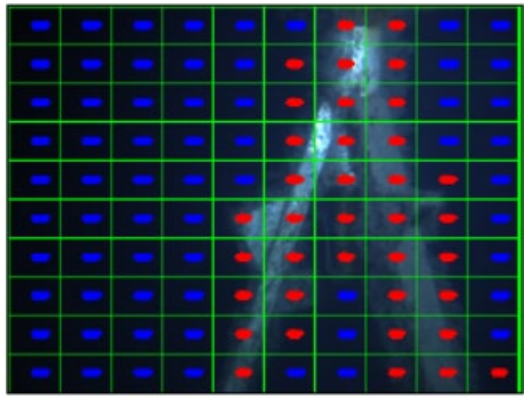


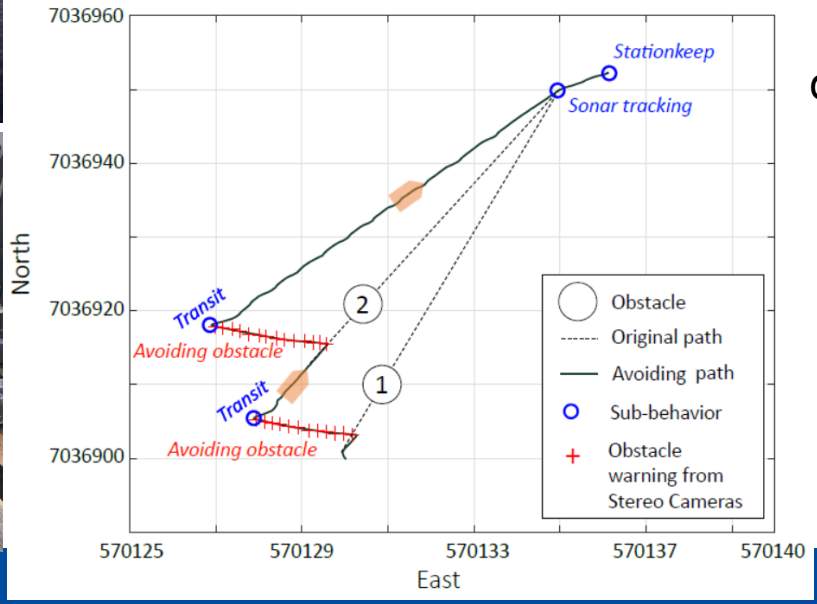
- ROV
  - IMU (MTi)
  - Pressure
  - DVL
- Surface Vessel
  - HiPaP
  - IMU
  - GPS
  - Control system computers and operator stations

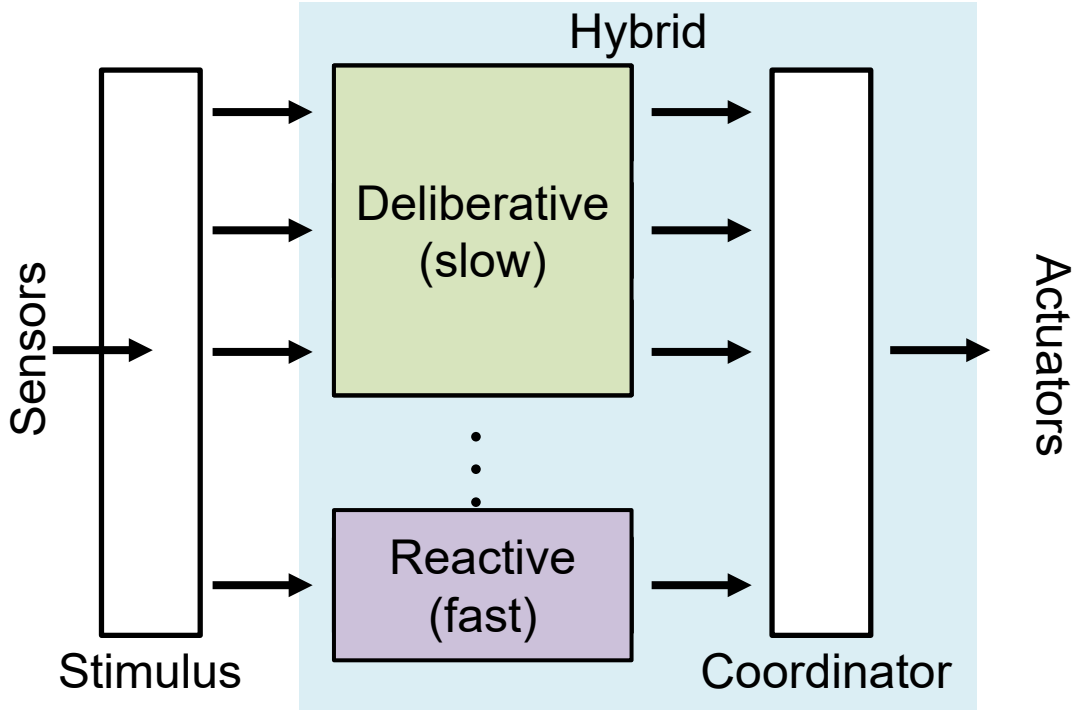
Autonomous mission: approach and localization of SOI



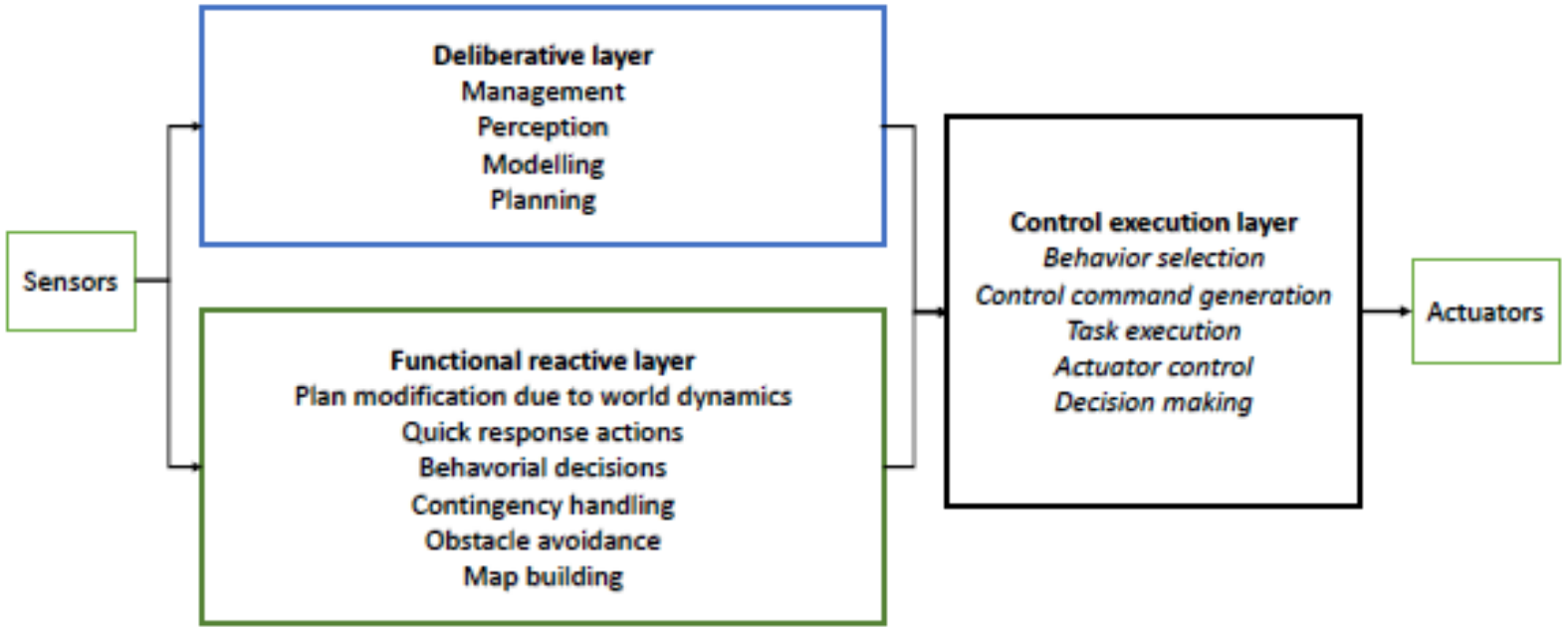
ROV autonomy







Hybrid control layer



AUV tendering

- Communication
- Navigation
- Cold and boring

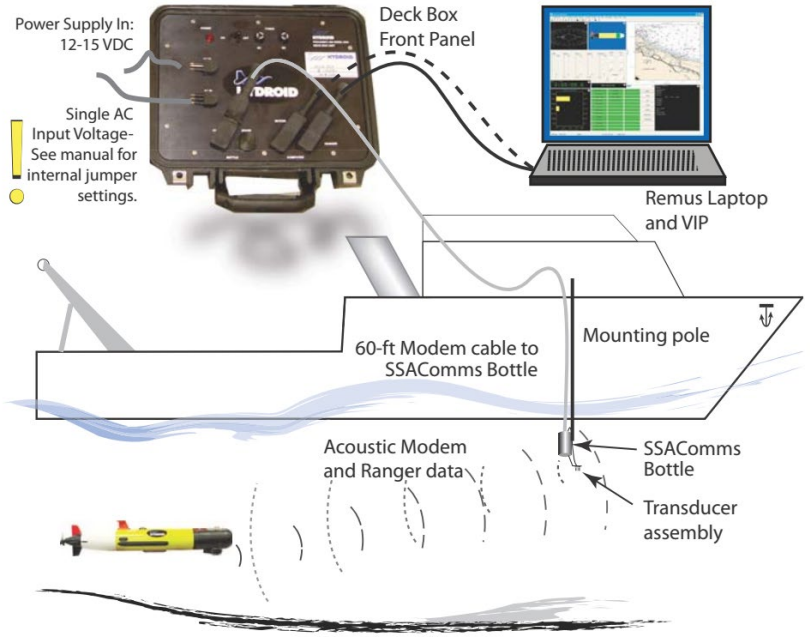
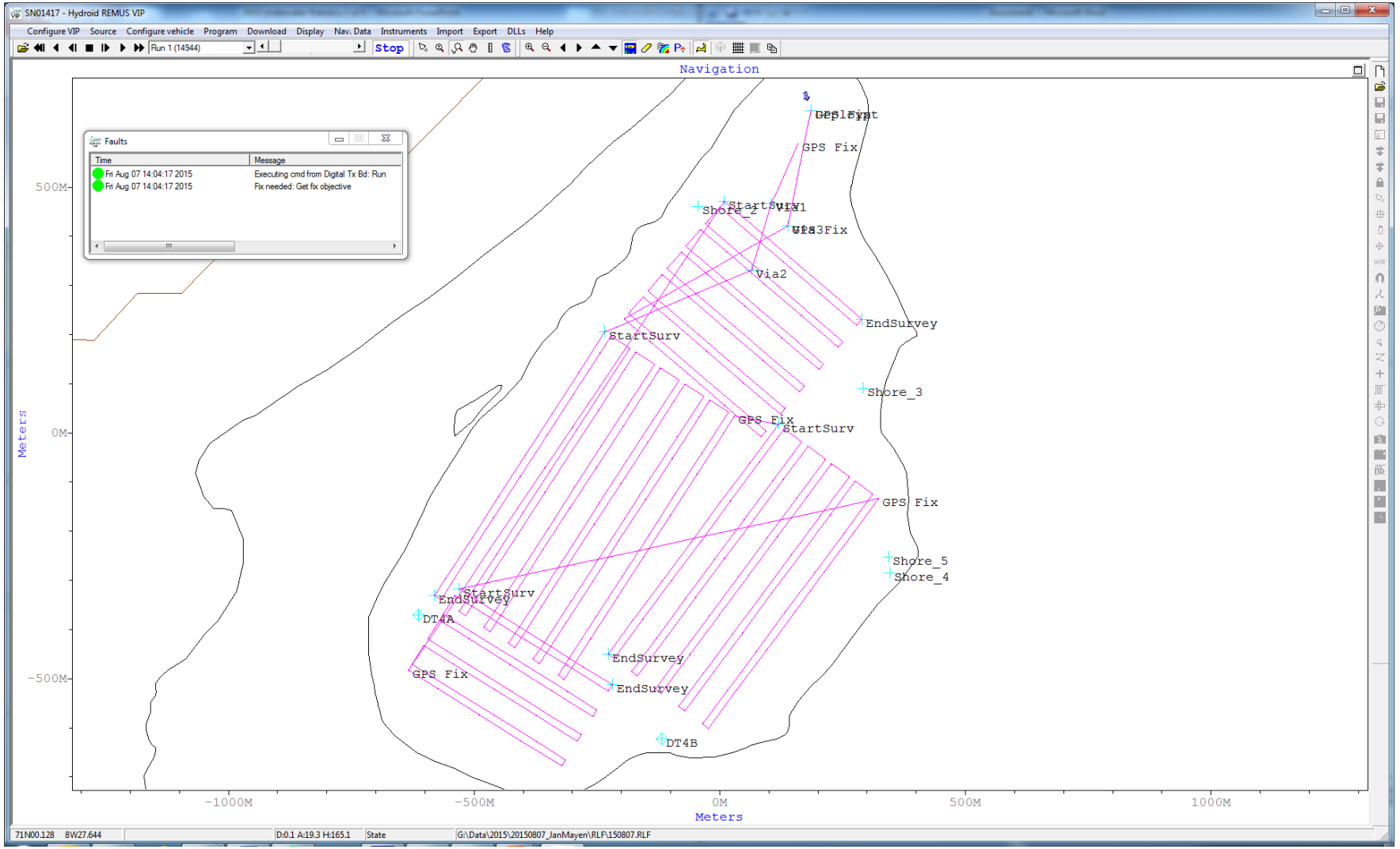


Figure from Kongsberg Hydroid

State of the art AUV REMUS



Hugin AUV - tendering

Very expensive and boring!

Collaborating unmanned vehicles

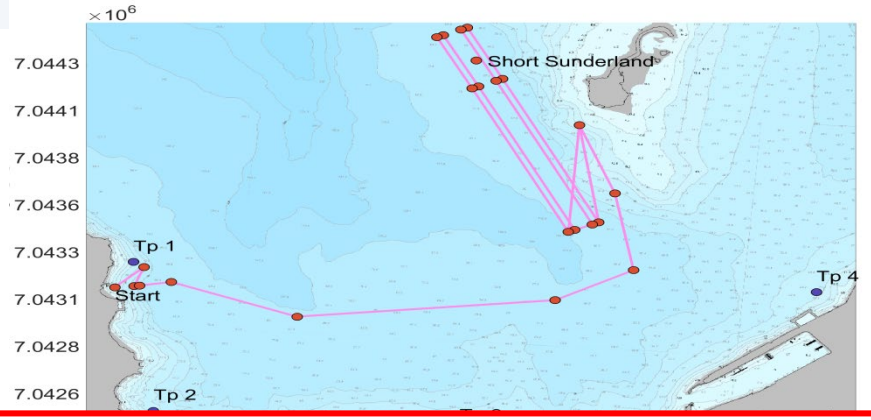
Motivation

- Establishing a continuous communication link is challenging.
- Navigation accuracy of AUV may degrade over time.
- Coordination of unmanned vehicles require relay of data.

Courtesy: UAV-lab, NTNU

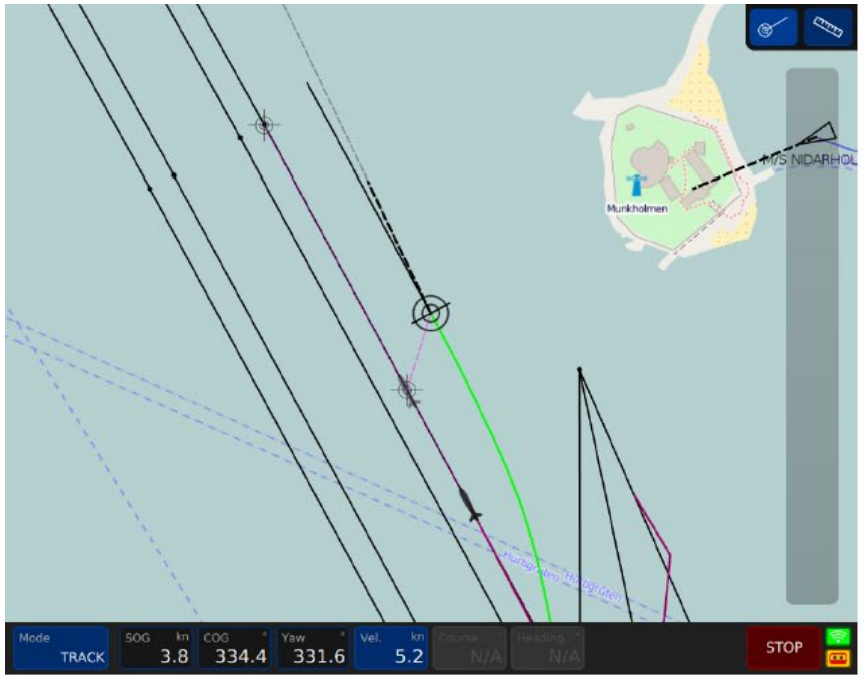
Courtesy: Maritime Robotics AS

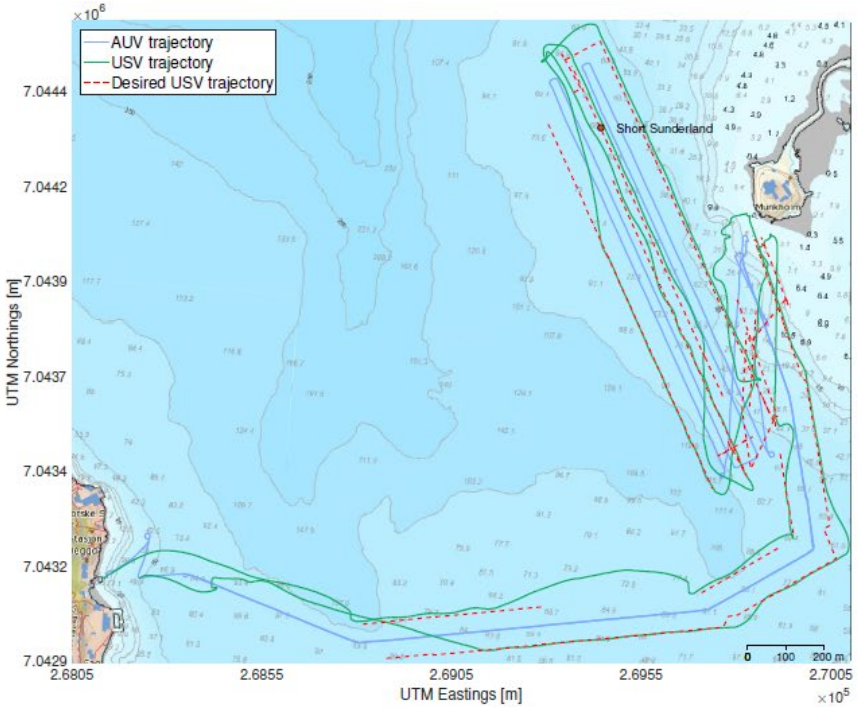
Courtesy: Øyvind Ødegård, AMOS, NTNU

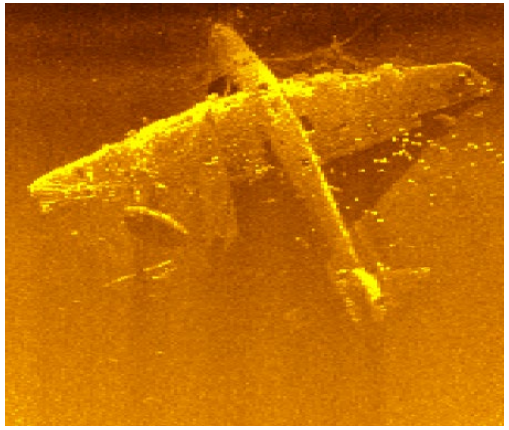


Norgren, Ludvigsen, Ingebretsen and Hovstein (2015) Tracking and remote monitoring of an autonomous underwater vehicle using an unmanned surface vehicle in the Trondheim fjord OCEANS, 2015 IEEE – Washington D.C. : October 19-22, IEEE conference proceedings.

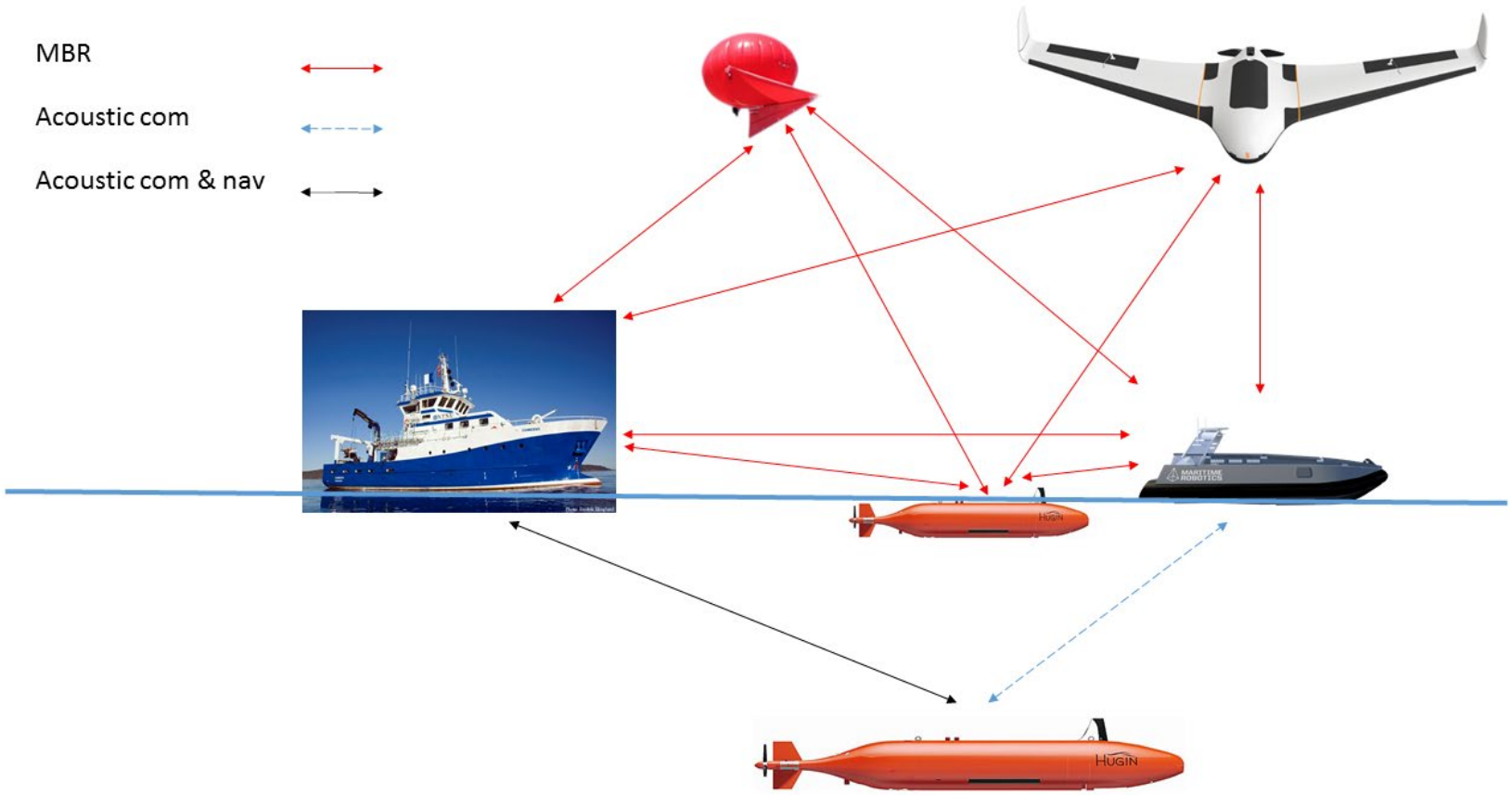
Results


Munkholmen
Mode
TRACK
SOG
3.8
COG
334.4
Yaw
331.6
Vel.
5.2
STOP


AUV trajectory
USV trajectory
Desired USV trajectory
Short Sunderland
UTM Northings [m]
UTM Eastings [m]
7.0444
7.0442
7.0439
7.0437
7.0434
7.0432
7.0429
2.6805
2.6855
2.6905
2.6955
2.7005



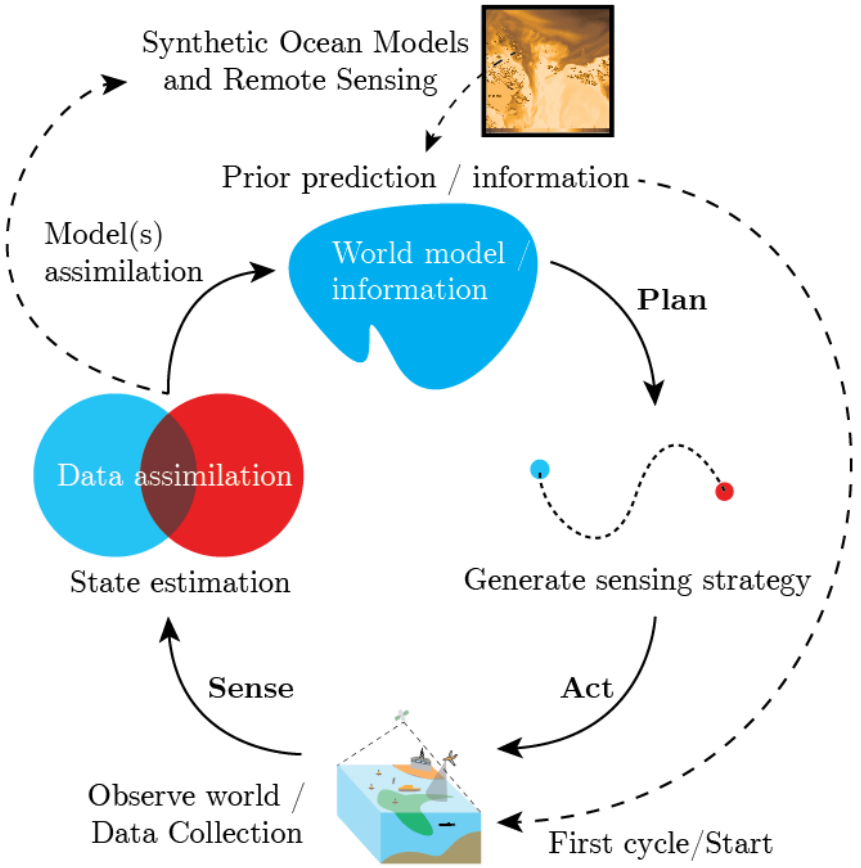
Network and communication



Closing the loop and covering the depths

Combine ocean models with robotic- and remote sensing in order to render an accurate representation of the ocean.

Use data-driven sampling to strategize sampling efforts.



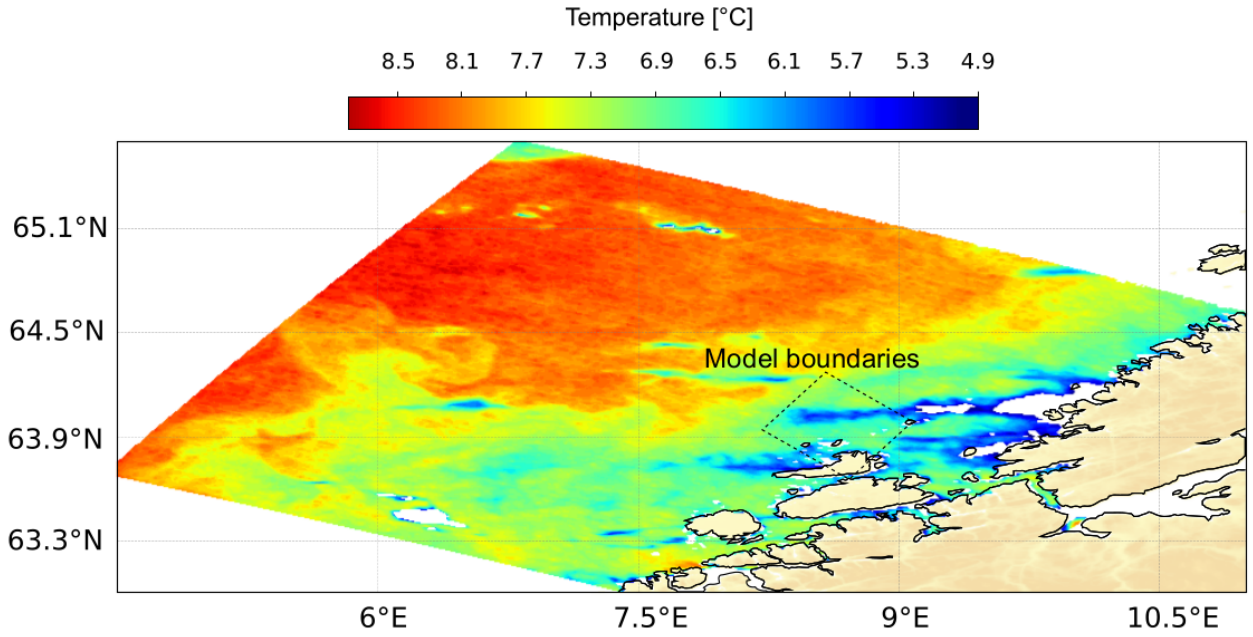
NTNU PhD Candidate Trygve Olav Fossum

L-AUV Harald

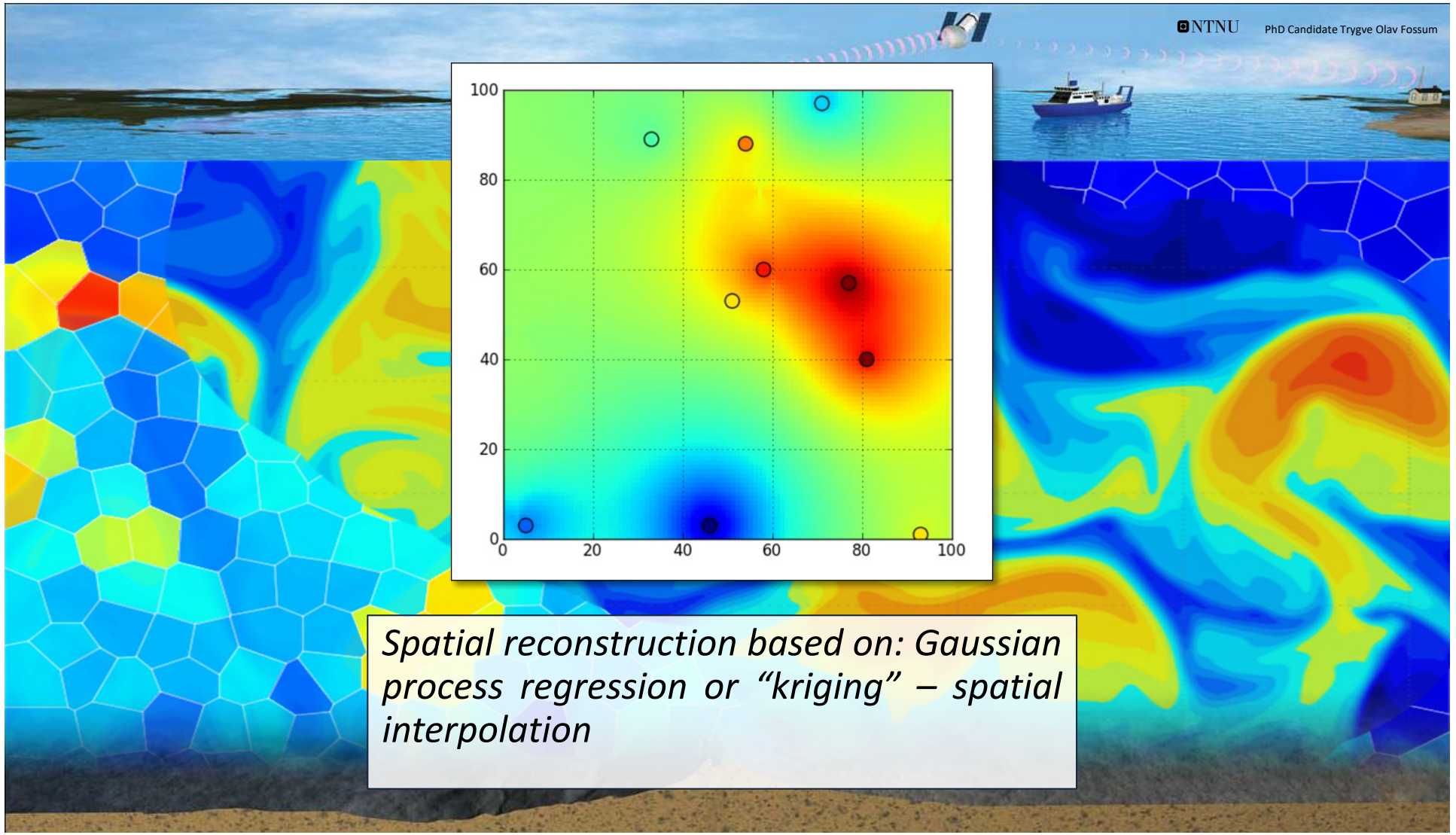
- Length: Harald: 2,4m
- Hull Diameter: 15cm
- Weight: Harald: 32,1kg

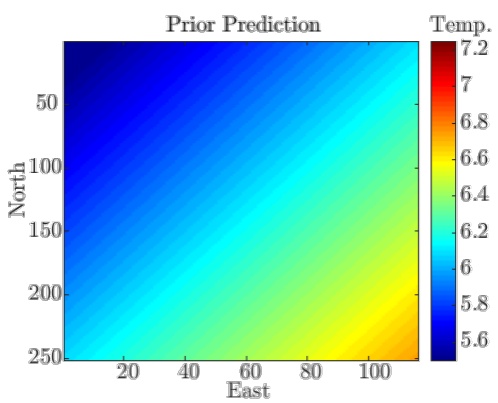
- Wireless Communications: Wi-Fi, GSM, Iridium SBD, Acoustic Modem, USBL
- Navigation: GPS, AHRS, Depth (pressure) Sensor, DVL
- Turbidity (Chl-a, cDOM, TSM), CTD, user CPU
- LSTS Software suite - DUNE

Adaptive AUV behavior

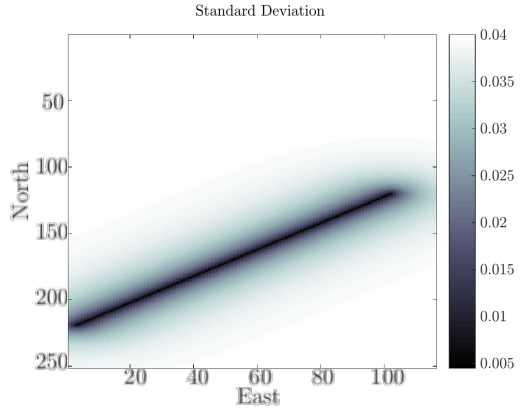


(a) Remote Sensing Satellite

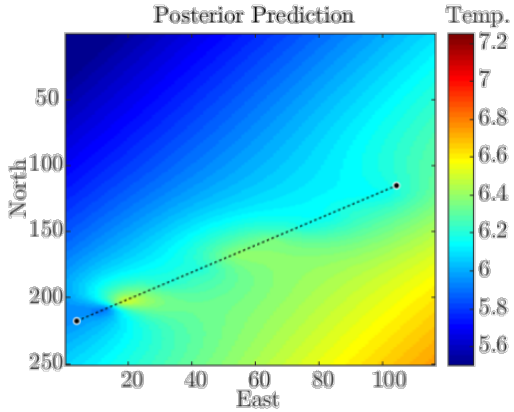

NTNU
PhD Candidate Trygve Olav Fossum
100
80
60
40
20
0
0
20
40
60
80
100
Spatial reconstruction based on: Gaussian process regression or “kriging” – spatial interpolation



(b) Prior GP.



(a) True temperature distribution.



(c) Posterior GP.

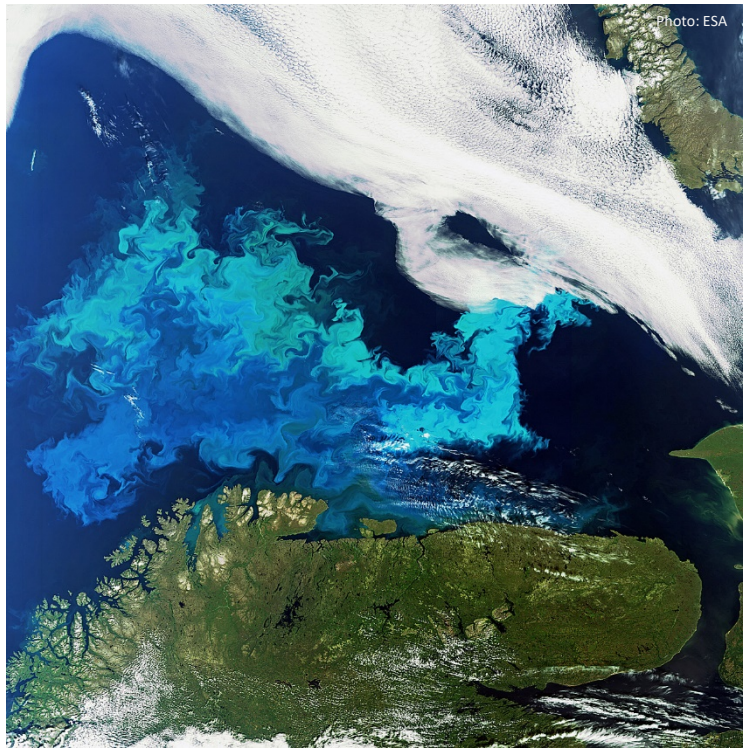
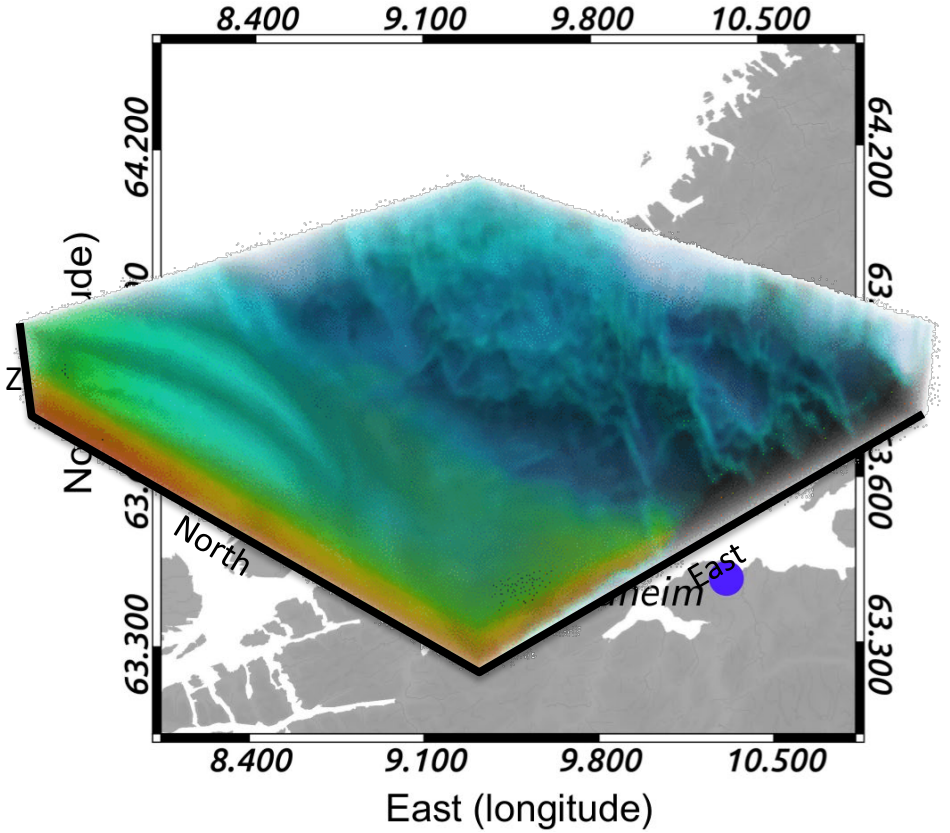


Photo: ESA


8.400
9.100
9.800
10.500
64.200
63.600
63.300
North
East
East (longitude)

Use an objective function to find locations that have high variance and gradients.

Conditional GPs based on data.

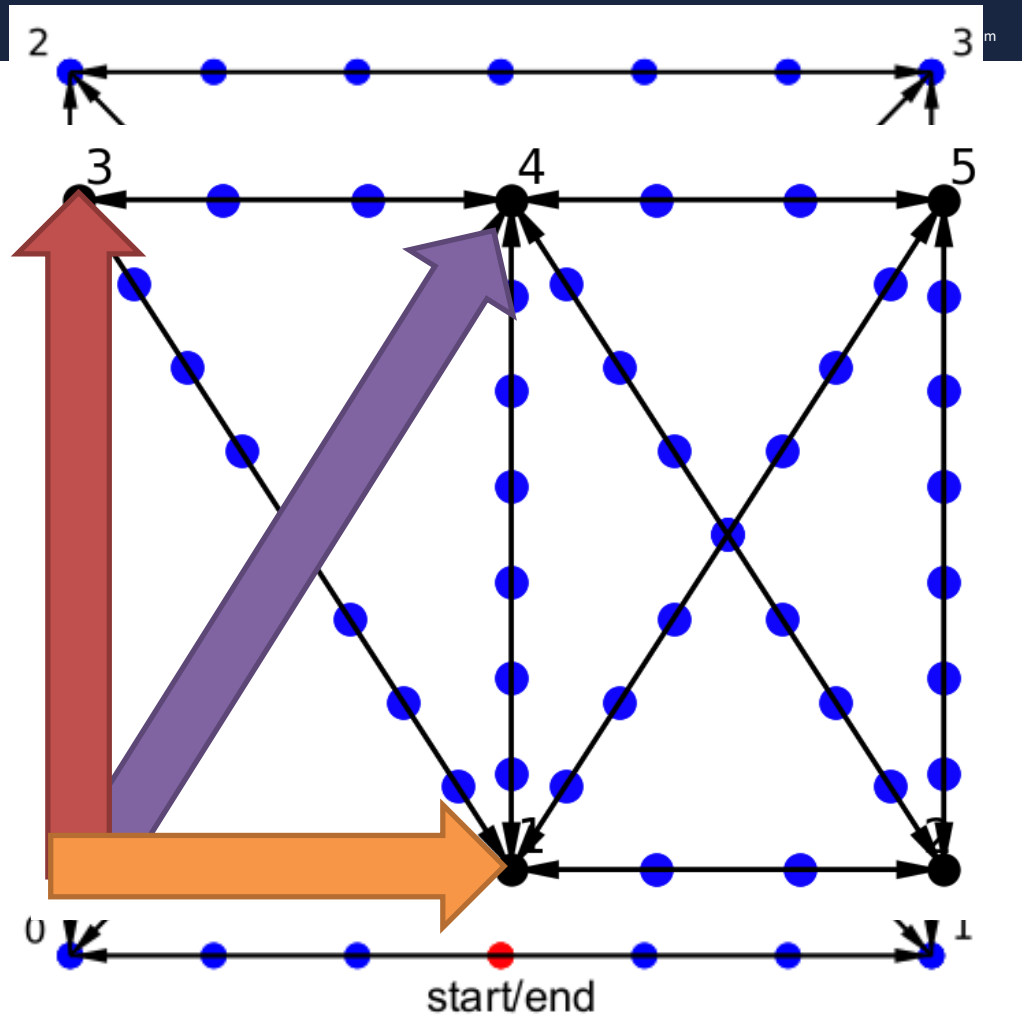
Myopic/Greedily chose where to go next.

$$f(\boldsymbol{d}_j) = \text{argmin}_{\boldsymbol{d}_j} \Big\{ \frac{1}{n} \text{trace}(\boldsymbol{P}_{t,\boldsymbol{d}_j}) - \frac{1}{n_k} \sum_{k=1}^{n_k} \nabla \boldsymbol{m}_t(\boldsymbol{d}_j(k), \boldsymbol{P}_{t,\boldsymbol{d}_j}) \Big\}.$$

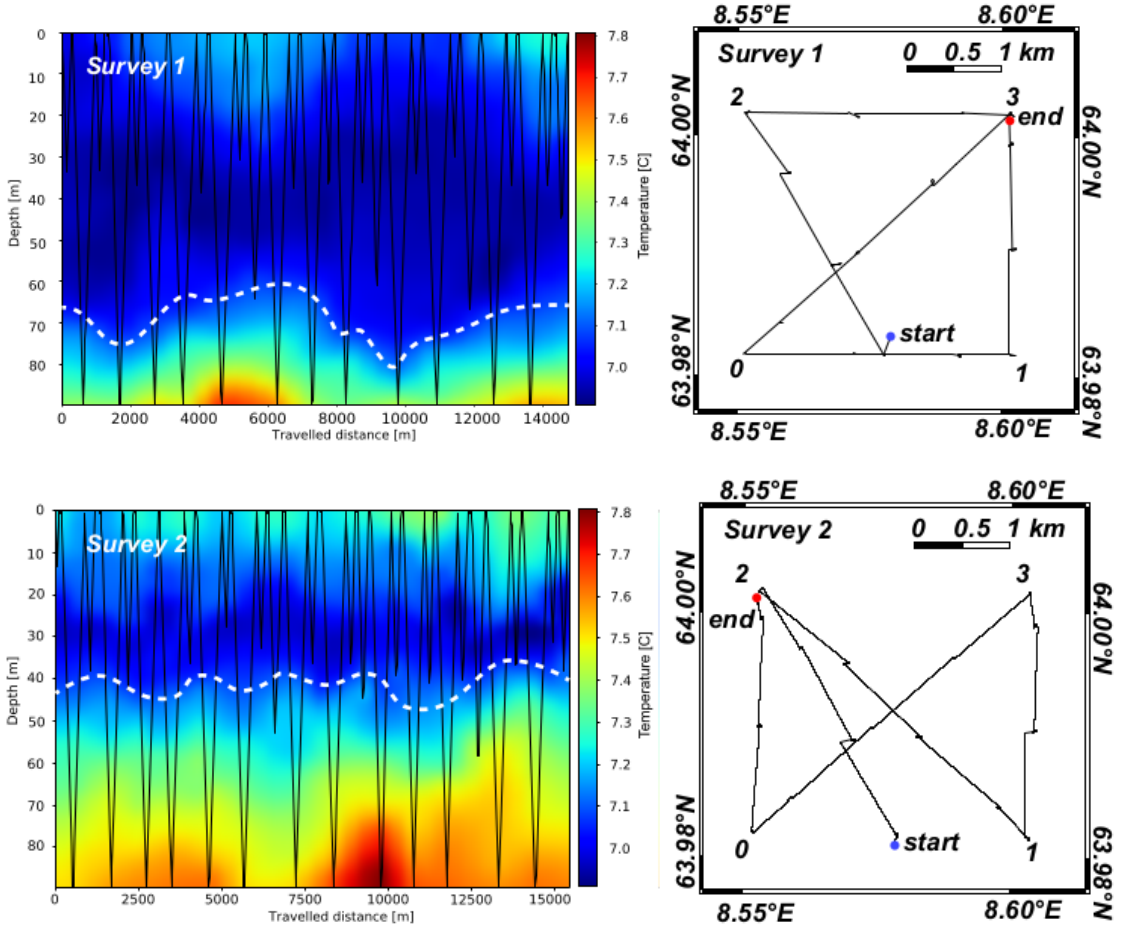
$\boldsymbol{P}_{t,\boldsymbol{d}_j}$ *is the covariance matrix*

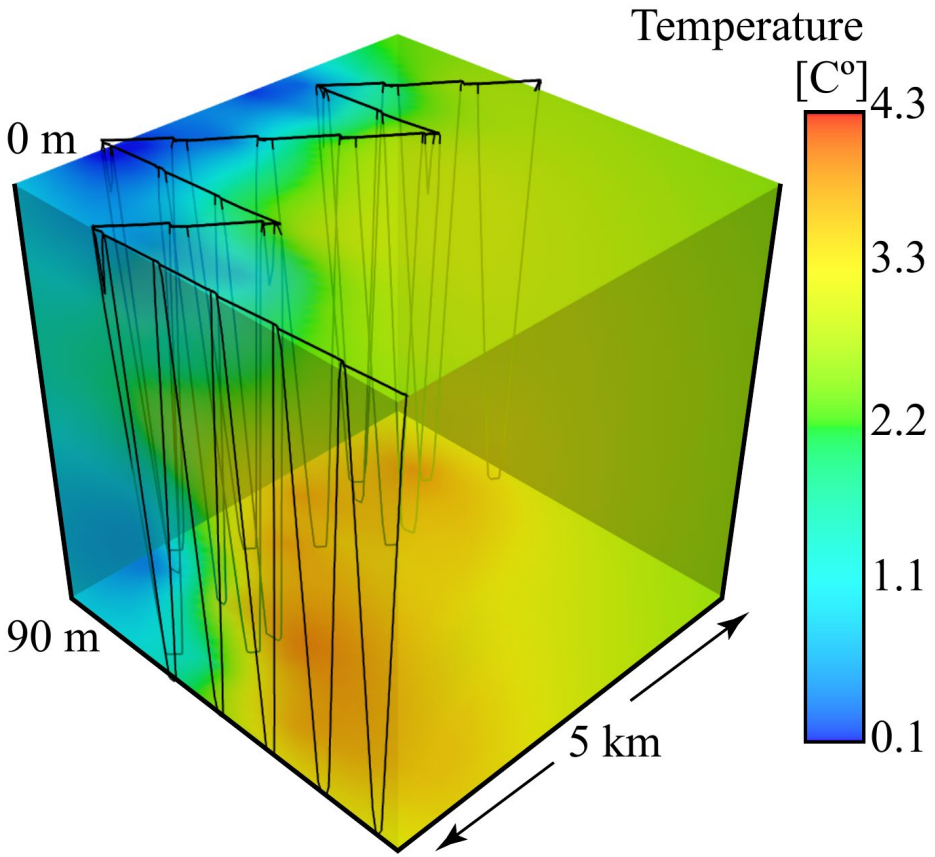
$\boldsymbol{m}_t$ *is the mean*

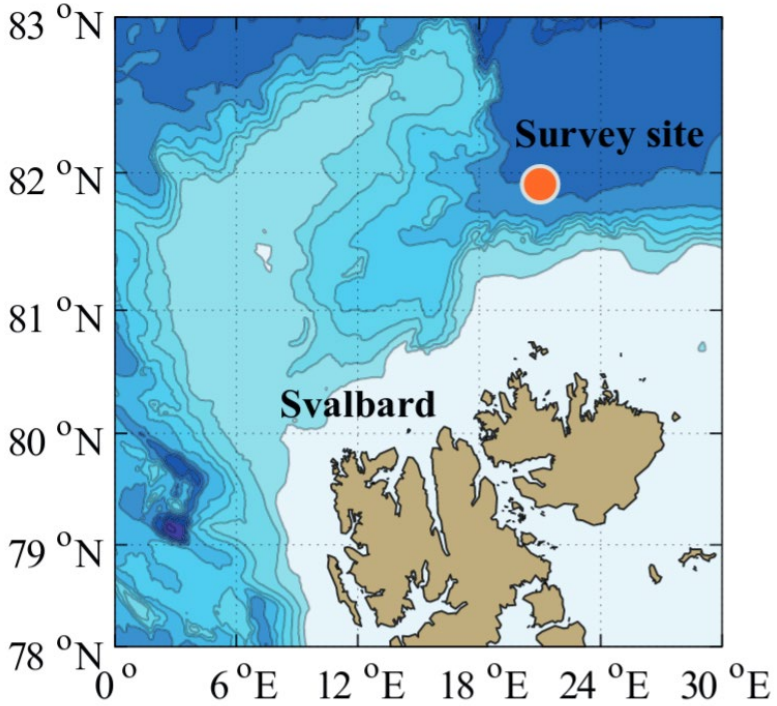
$\boldsymbol{d}_j$ *is the data*



Research project: ENTiCE







NTNU PhD Candidate Trygve Olav Fossum

Learning objectives autonomy

- Control architecture for underwater vehicles
- Levels of autonomy
- Architectures
  - Reactive and deliberative systems
- Behaviors
- Research examples

More reading

- Handbook of Robotics (*Siciliano 2008*)
- Marine Robot Autonomy (*Seto 2013*)