Accelerated Marine Vehicle Autonomy, Sensing, and Communications

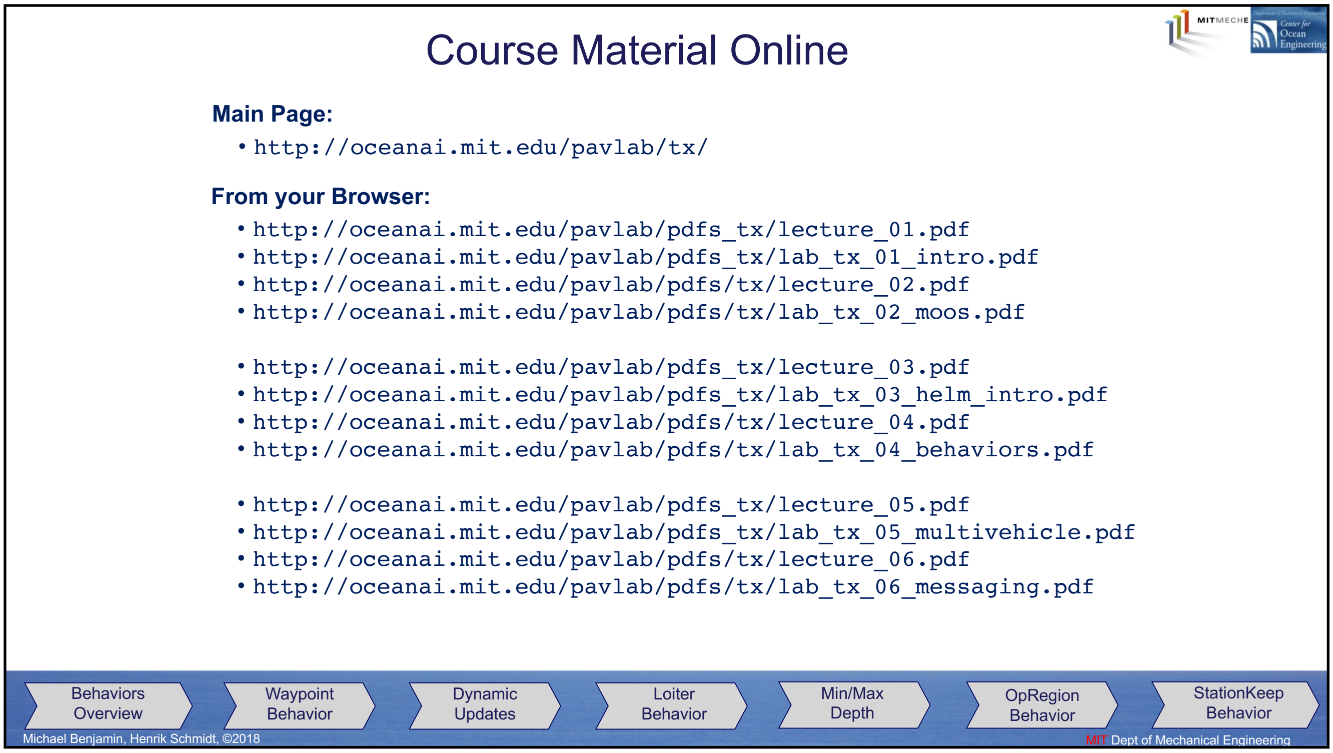
Spring Semester 2019
2.014 Autonomy Mini-course

A Deeper Dive Into Behaviors

Web: http://oceanai.mit.edu/pavlab/textron

Mike Benjamin, mikerb@mit.edu
Henrik Schmidt, henrik@mit.edu

Accelerated Marine Autonomy – "A Deeper Dive Into Behaviors"



Course Material Online

Main Page:

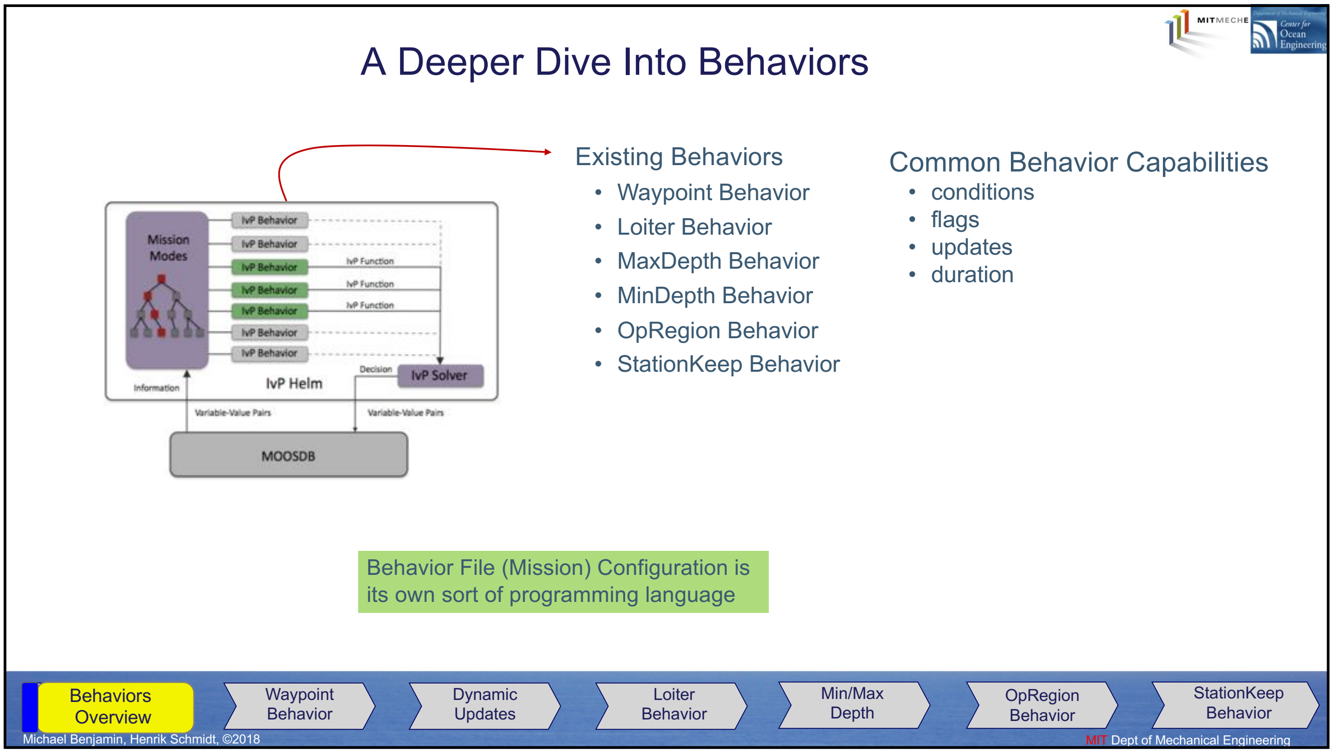
- `http://oceanai.mit.edu/pavlab/tx/`

From your Browser:

- `http://oceanai.mit.edu/pavlab/pdfs_tx/lecture_01.pdf`
- `http://oceanai.mit.edu/pavlab/pdfs_tx/lab_tx_01_intro.pdf`
- `http://oceanai.mit.edu/pavlab/pdfs/tx/lecture_02.pdf`
- `http://oceanai.mit.edu/pavlab/pdfs/tx/lab_tx_02_moos.pdf`

- `http://oceanai.mit.edu/pavlab/pdfs_tx/lecture_03.pdf`
- `http://oceanai.mit.edu/pavlab/pdfs_tx/lab_tx_03_helm_intro.pdf`
- `http://oceanai.mit.edu/pavlab/pdfs/tx/lecture_04.pdf`
- `http://oceanai.mit.edu/pavlab/pdfs/tx/lab_tx_04_behaviors.pdf`

- `http://oceanai.mit.edu/pavlab/pdfs_tx/lecture_05.pdf`
- `http://oceanai.mit.edu/pavlab/pdfs_tx/lab_tx_05_multivehicle.pdf`
- `http://oceanai.mit.edu/pavlab/pdfs/tx/lecture_06.pdf`
- `http://oceanai.mit.edu/pavlab/pdfs/tx/lab_tx_06_messaging.pdf`



A Deeper Dive Into Behaviors

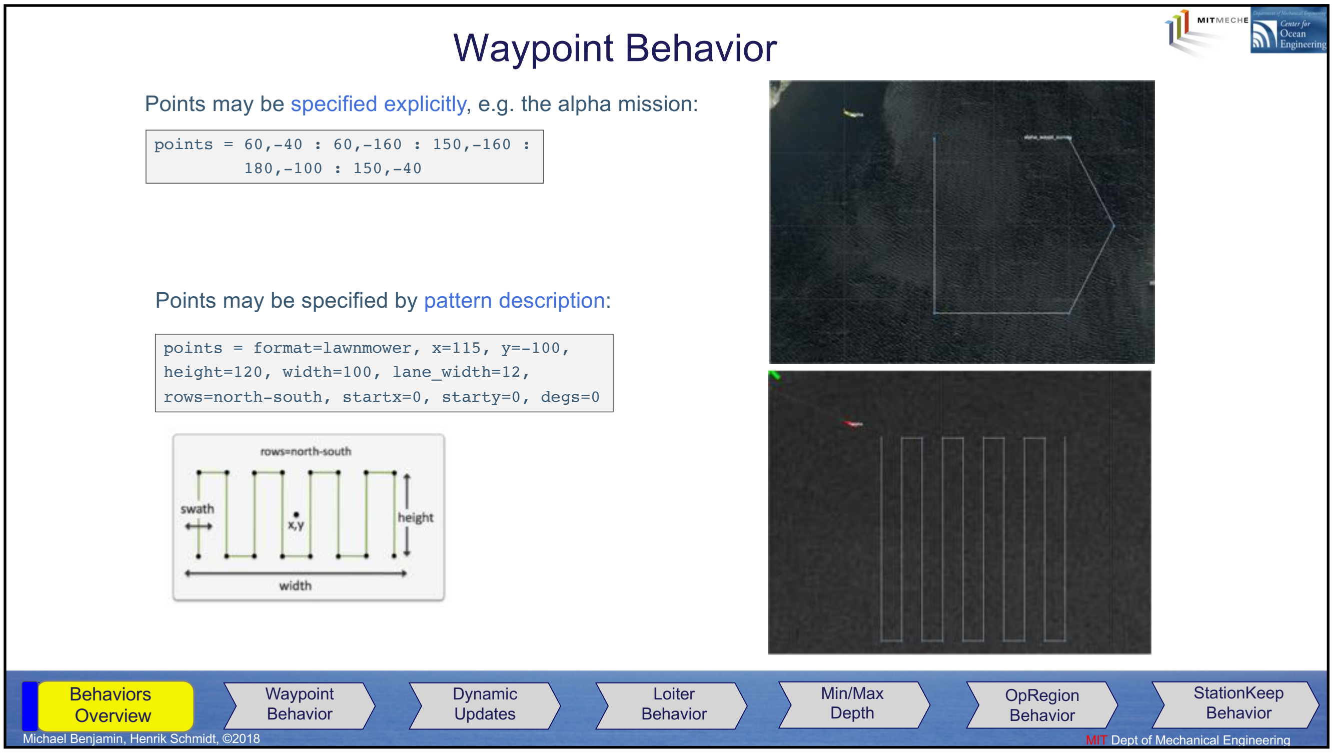
Existing Behaviors

- Waypoint Behavior
- Loiter Behavior
- MaxDepth Behavior
- MinDepth Behavior
- OpRegion Behavior
- StationKeep Behavior

Common Behavior Capabilities

- conditions
- flags
- updates
- duration

Behavior File (Mission) Configuration is its own sort of programming language



Waypoint Behavior

Points may be specified explicitly, e.g. the alpha mission:

```
points = 60,-40 : 60,-160 : 150,-160 :
         180,-100 : 150,-40
```

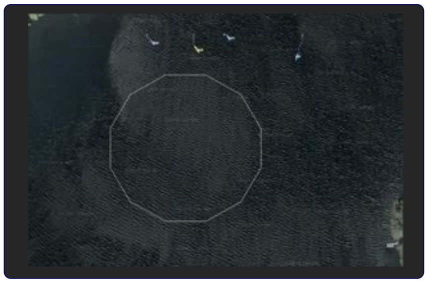
Points may be specified by pattern description:

```
points = format=lawnmower, x=115, y=-100,
height=120, width=100, lane_width=12,
rows=north-south, startx=0, starty=0, degs=0
```

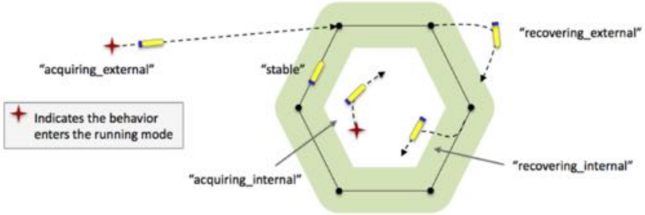
Loiter Behavior

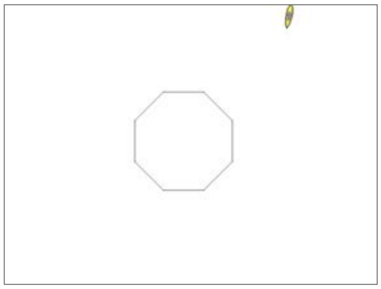
Points specified by may be convex polygon

```
polygon = radial::x=75,y=-75,radius=50,pts=12
```



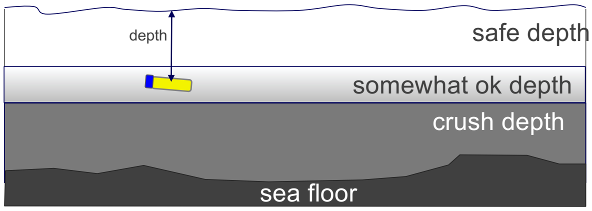
Loiter entry and recover is robust to disruptions



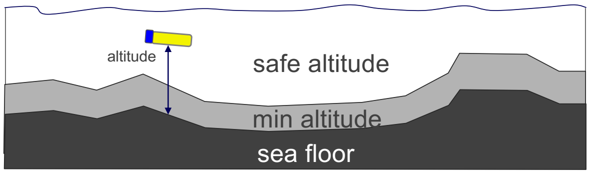


Min Altitude / Max Depth Behaviors

- **MaxDepth behavior** will disallow a depth command below critical depth.
- Near-critical depths are ranked poorly but could be allowed if other behaviors need to go deep.

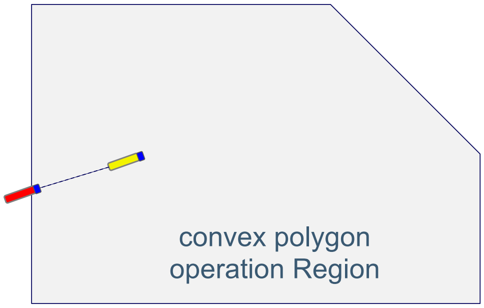


- **MinAltitude behavior** will disallow depths with low altitude to the sea floor
- Near-critical altitudes are ranked poorly but could be allowed if other behaviors need to go deep.



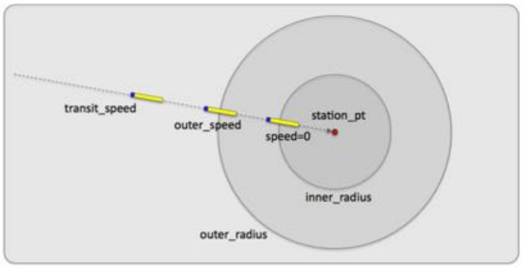
OpRegion Behavior

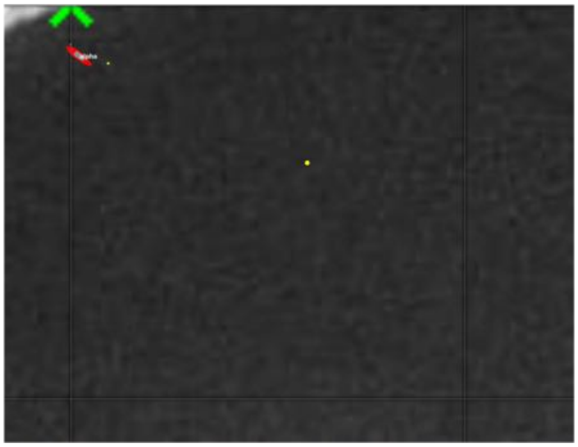
- **OpRegion behavior** has a convex polygon region.
- If the vehicle goes outside this region, a vehicle all-stop is issued.
- Status posts are made indicating range/time to exiting the region. To allow corrective actions to be initiated



StationKeep Behavior

- **StationKeep behavior** keeps a vehicle on station defined by a point
- It can be set to continuously adjust
- It can be set to periodically adjust while drifting during inactivity

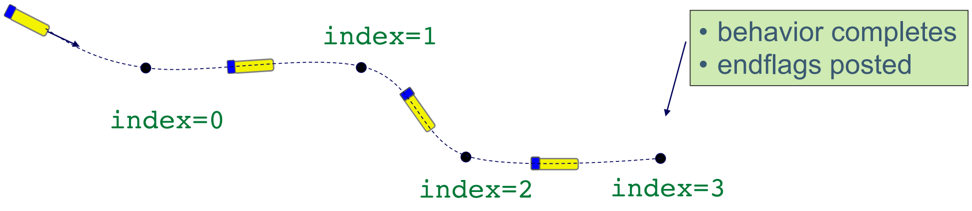




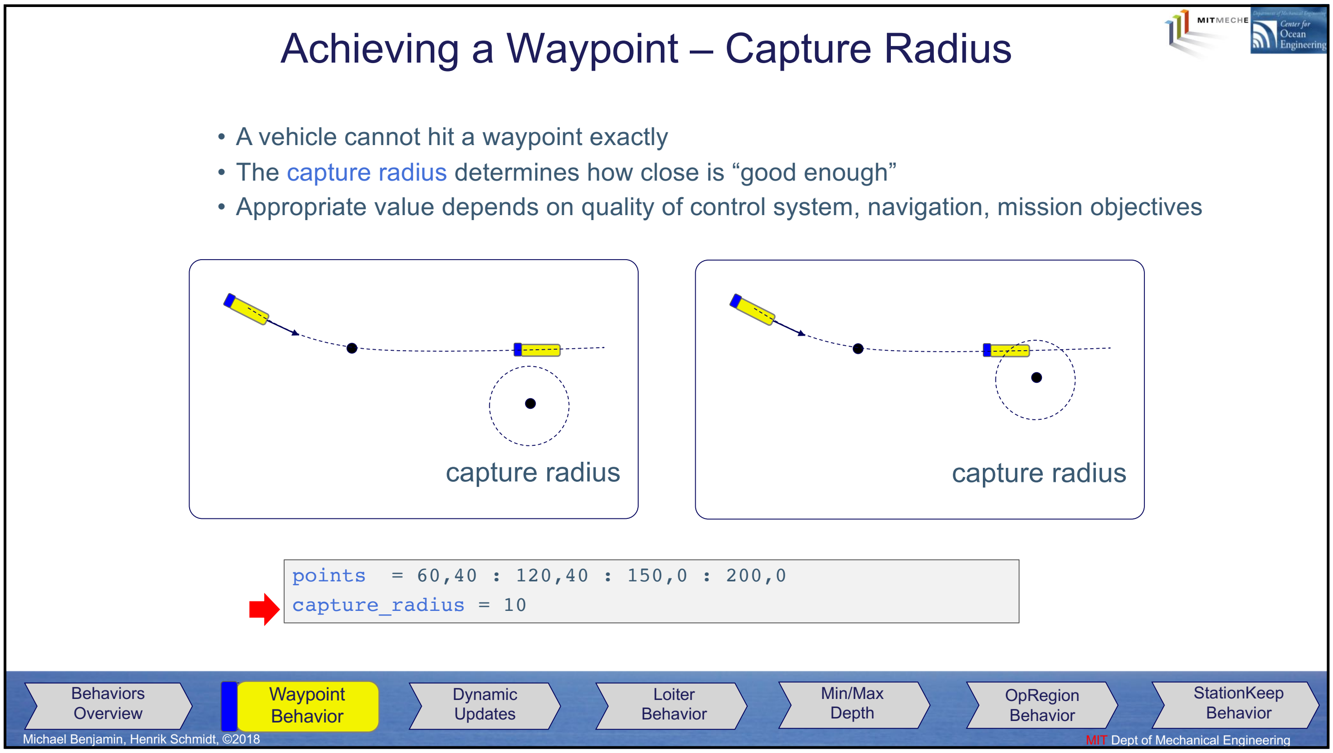
The Waypoint Behavior (Deeper Dive)

Traversing Waypoints

- The set of waypoints, will be traversed in order. Each waypoint has an index
- Upon each waypoint, a waypoint flag may be posted, if configured in the mission
- The behavior will completes when it has visited all waypoints



```
points  = 60,40 : 120,40 : 150,0 : 200,0
endflag = RETURN=true
wptflag = MEASURE=true
```



Achieving a Waypoint – Capture Radius

- A vehicle cannot hit a waypoint exactly
- The capture radius determines how close is “good enough”
- Appropriate value depends on quality of control system, navigation, mission objectives

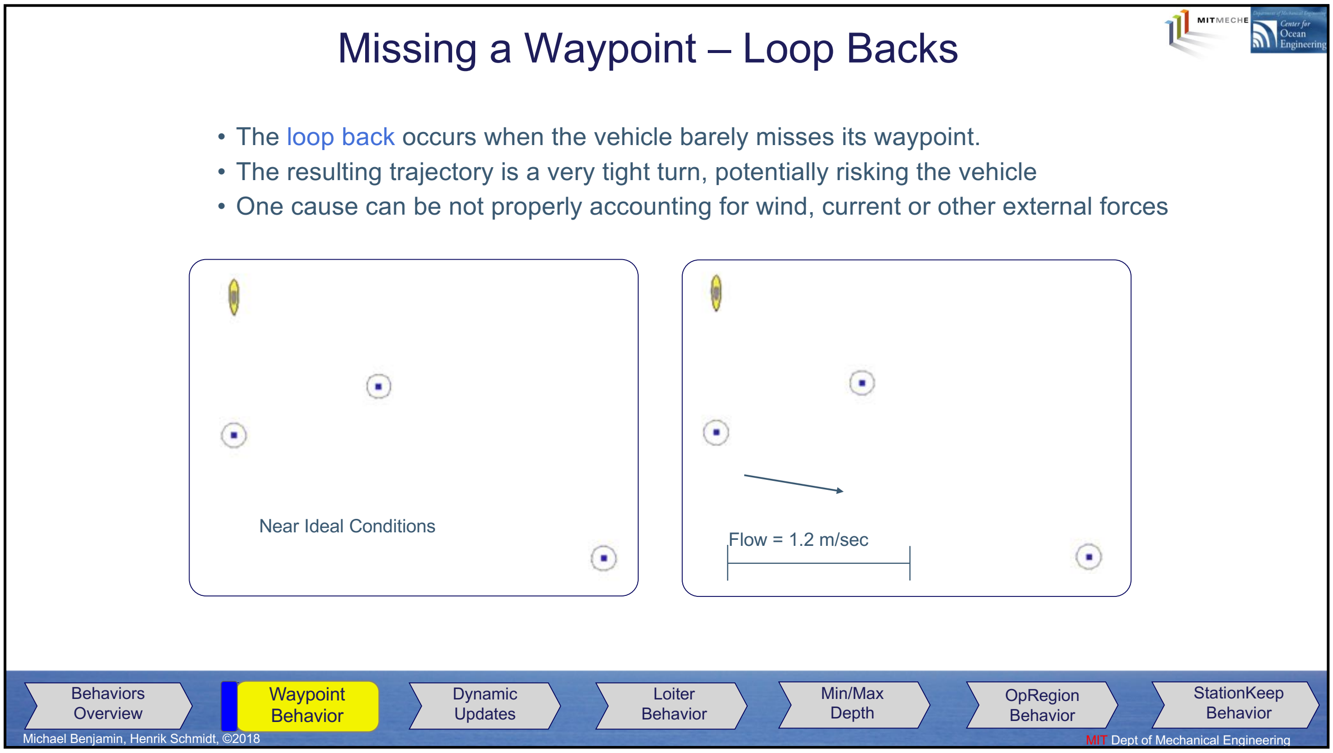
capture radius
capture radius

```
points   = 60,40 : 120,40 : 150,0 : 200,0
capture_radius = 10
```

Behaviors Overview | Waypoint Behavior | Dynamic Updates | Loiter Behavior | Min/Max Depth | OpRegion Behavior | StationKeep Behavior

Michael Benjamin, Henrik Schmidt, ©2018

MIT Dept of Mechanical Engineering



Missing a Waypoint – Loop Backs

- The loop back occurs when the vehicle barely misses its waypoint.
- The resulting trajectory is a very tight turn, potentially risking the vehicle
- One cause can be not properly accounting for wind, current or other external forces

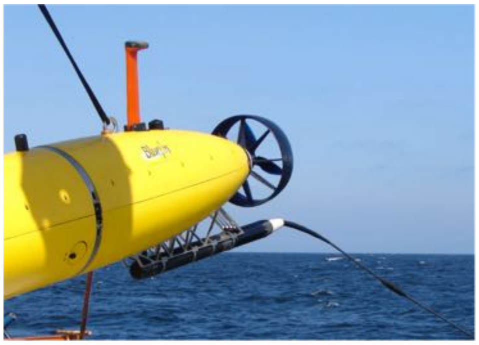
Near Ideal Conditions
Flow = 1.2 m/sec

Behaviors Overview | Waypoint Behavior | Dynamic Updates | Loiter Behavior | Min/Max Depth | OpRegion Behavior | StationKeep Behavior

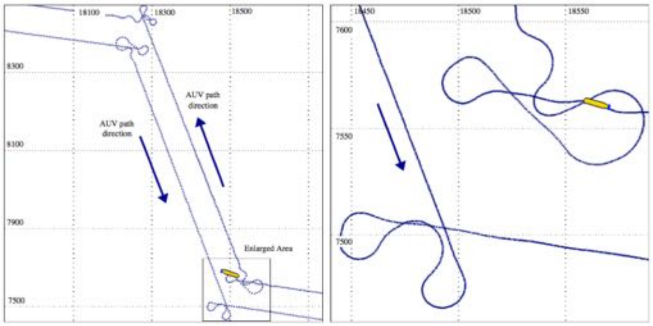
Michael Benjamin, Henrik Schmidt, ©2018

MIT Dept of Mechanical Engineering

Monterey Bay California 2006

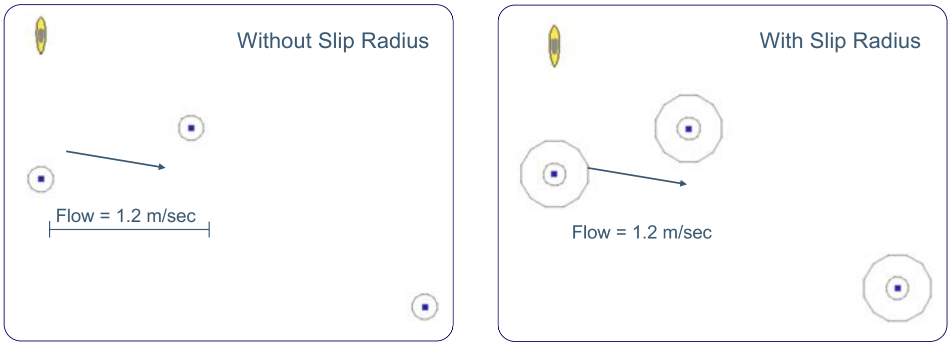


Adverse Affects of Loop-Backs



Achieving a Waypoint – Slip Radius

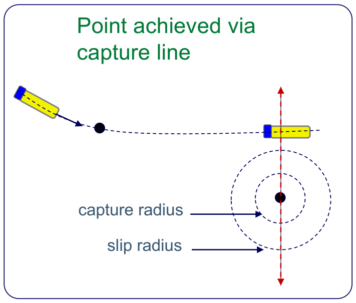
- Larger capture radius reduces loop-backs, but means you “arrive” sooner
- The slip radius allows the capture radius to be missed, but still achieve the waypoint
- If the vehicle enters the slip radius, and begins to exit, we say the point is achieved



```
points   = 60,40 : 120,40 : 150,0 : 200,0
capture_radius = 10
slip_radius = 25
```

Achieving a Waypoint – Capture Line

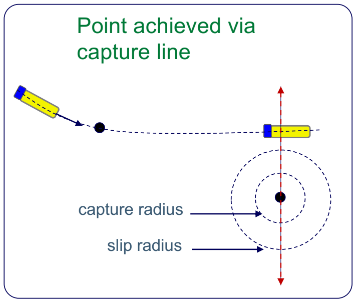
- A capture line is an additional capture criteria, when robot crosses the line
- Line is perpendicular to the line between the waypoint and the point when the robot begins striving for that point

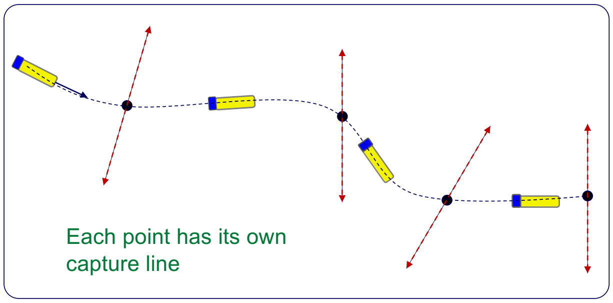


```
points   = 60,40 : 120,40 : 150,0 : 200,0
capture_radius = 10
slip_radius = 25
capture_line = true
```

Achieving a Waypoint – Capture Line

- A capture line is an additional capture criteria, when robot crosses the line
- Line is perpendicular to the line between the waypoint and the point when the robot begins striving for that point

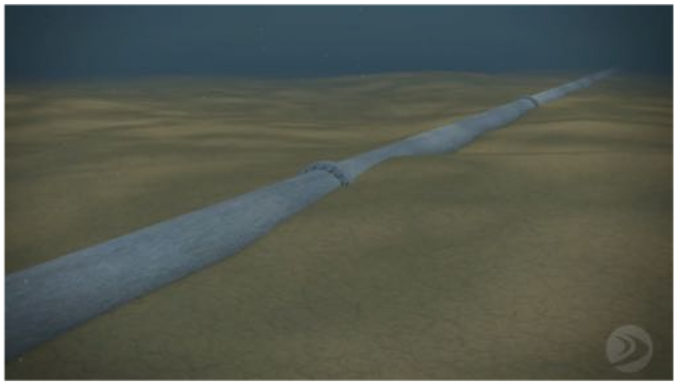


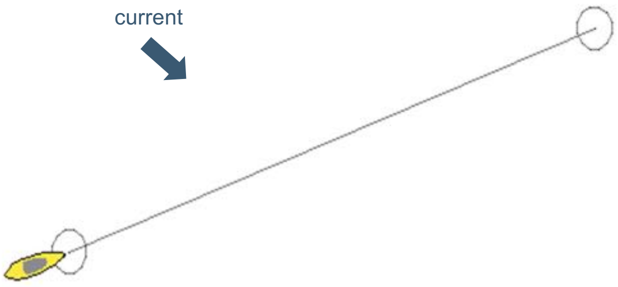


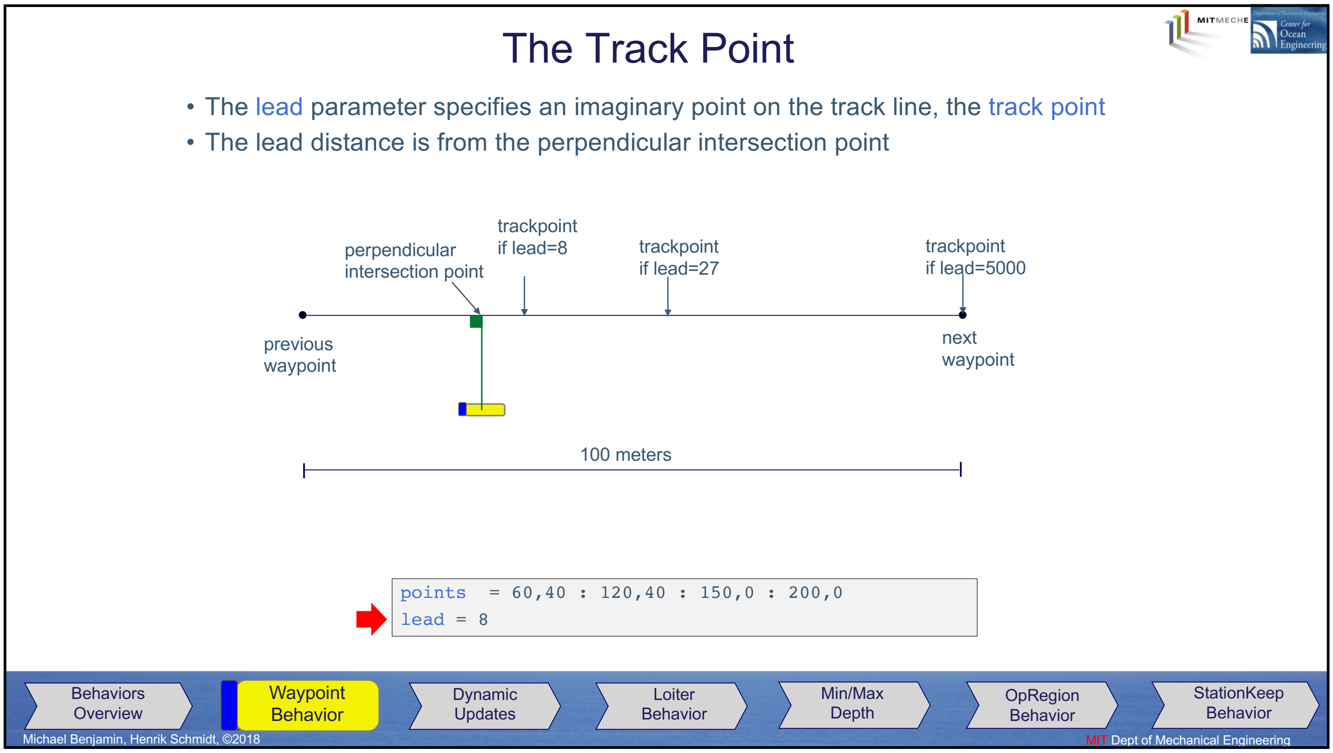
```
points  = 60,40 : 120,40 : 150,0 : 200,0
capture_radius = 10
slip_radius = 25
capture_line = true
```

Track-line Following

- In some missions, a vehicle needs to follow a track-line, for optimal sensing
- This may be hard due to vehicle dynamics
- The environment (current, wind) may also cause problems



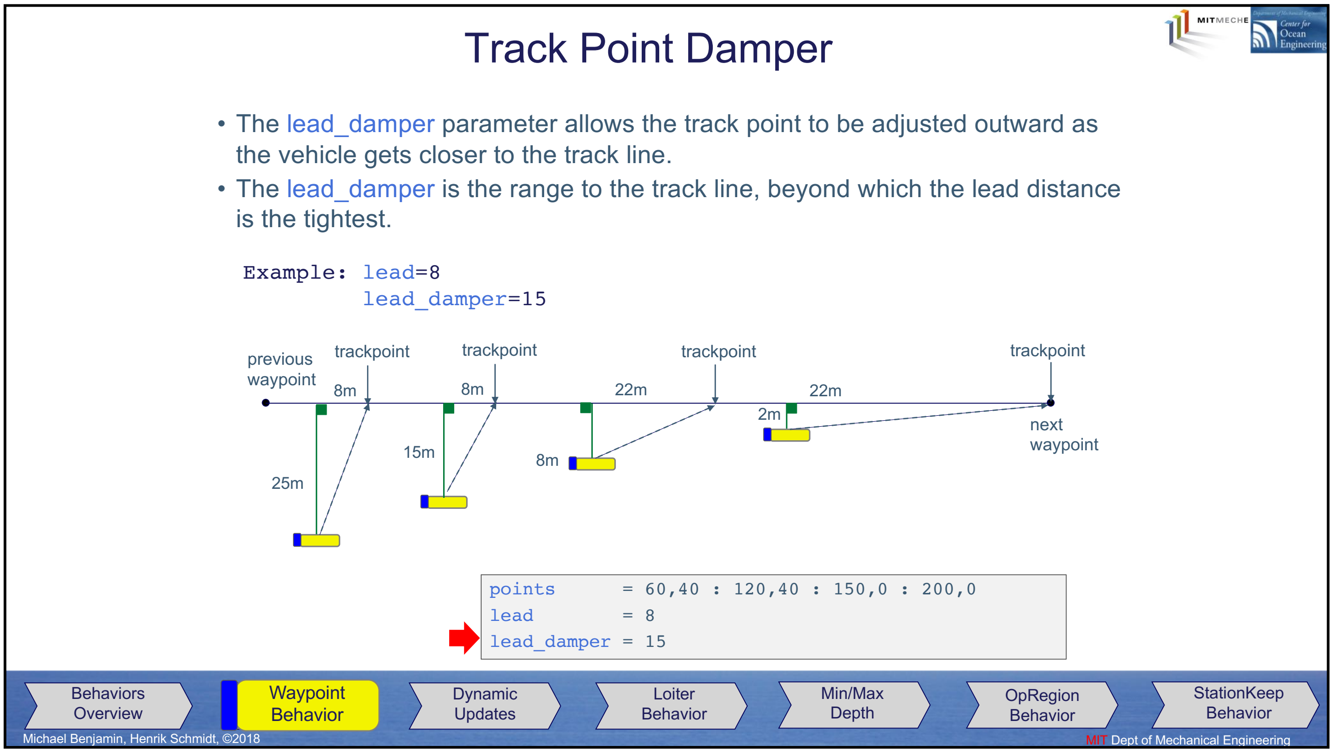




The Track Point

- The lead parameter specifies an imaginary point on the track line, the track point
- The lead distance is from the perpendicular intersection point

```
points   = 60,40 : 120,40 : 150,0 : 200,0
lead = 8
```



Track Point Damper

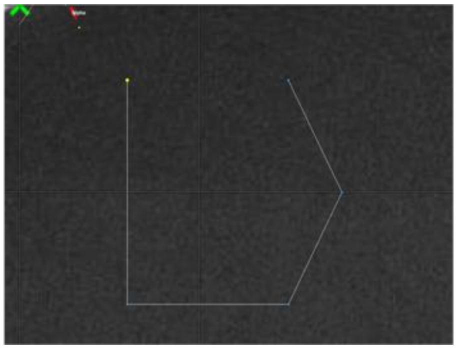
- The lead_damper parameter allows the track point to be adjusted outward as the vehicle gets closer to the track line.
- The lead_damper is the range to the track line, beyond which the lead distance is the tightest.

```
Example: lead=8
         lead_damper=15
```

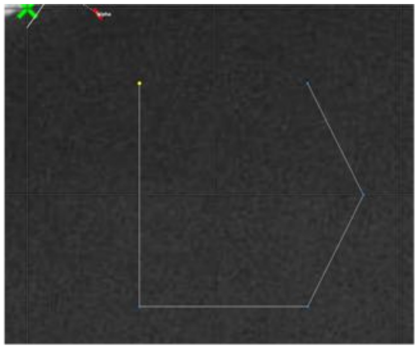
```
points      = 60,40 : 120,40 : 150,0 : 200,0
lead        = 8
lead_damper = 15
```

Alpha With and Without Track-Line

Alpha With Track-Line



Alpha Without Track-Line

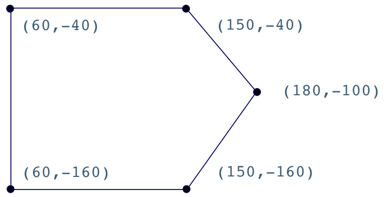


```
points   = 60,-40 : 60,-160 : 150,-160 : 180,-100 : 150,-40
capture_radius = 5
slip_radius = 15
lead = 8
```

Specifying Waypoints Explicitly

- Waypoints may be configured explicitly (as in the Alpha mission)

```
points   = 60,-40 : 60,-160 : 150,-160 : 180,-100 : 150,-40
```



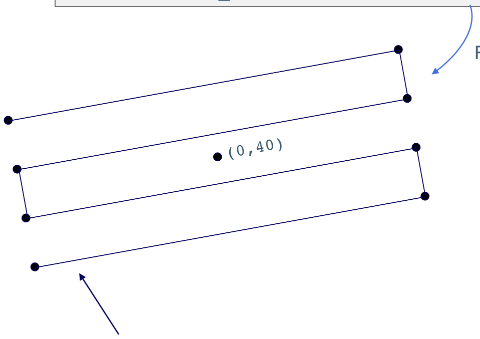
- Or simply a single point

```
point   = 60,-40
```

Waypoints as a Lawnmower Pattern

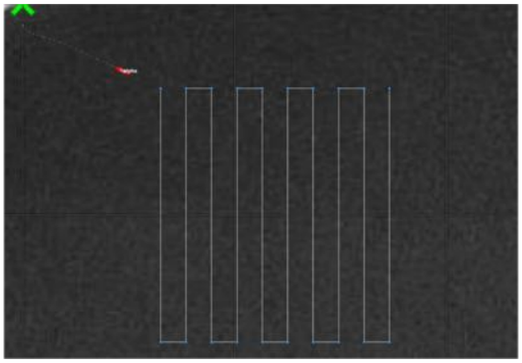
- Waypoints may be configured implicitly via lawnmower pattern parameters

```
points   = format=lawnmower, label=foxtrot, x=0, y=40, height=60, width=180,
           lane_width=15, rows=east-west, degs=45, startx=-20, starty=-300
```



Rotation specified

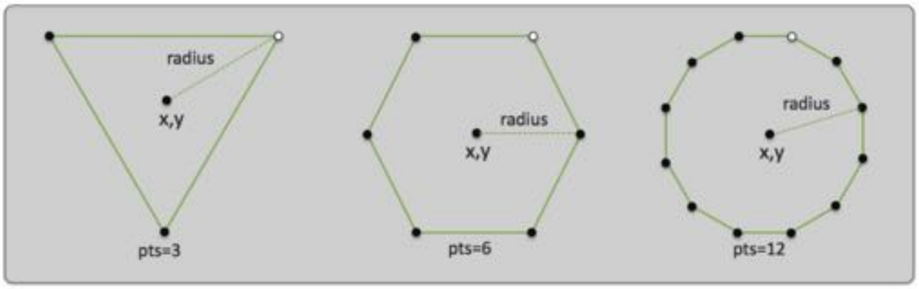
The first waypoint is the closest to the point given by `startx` and `starty`

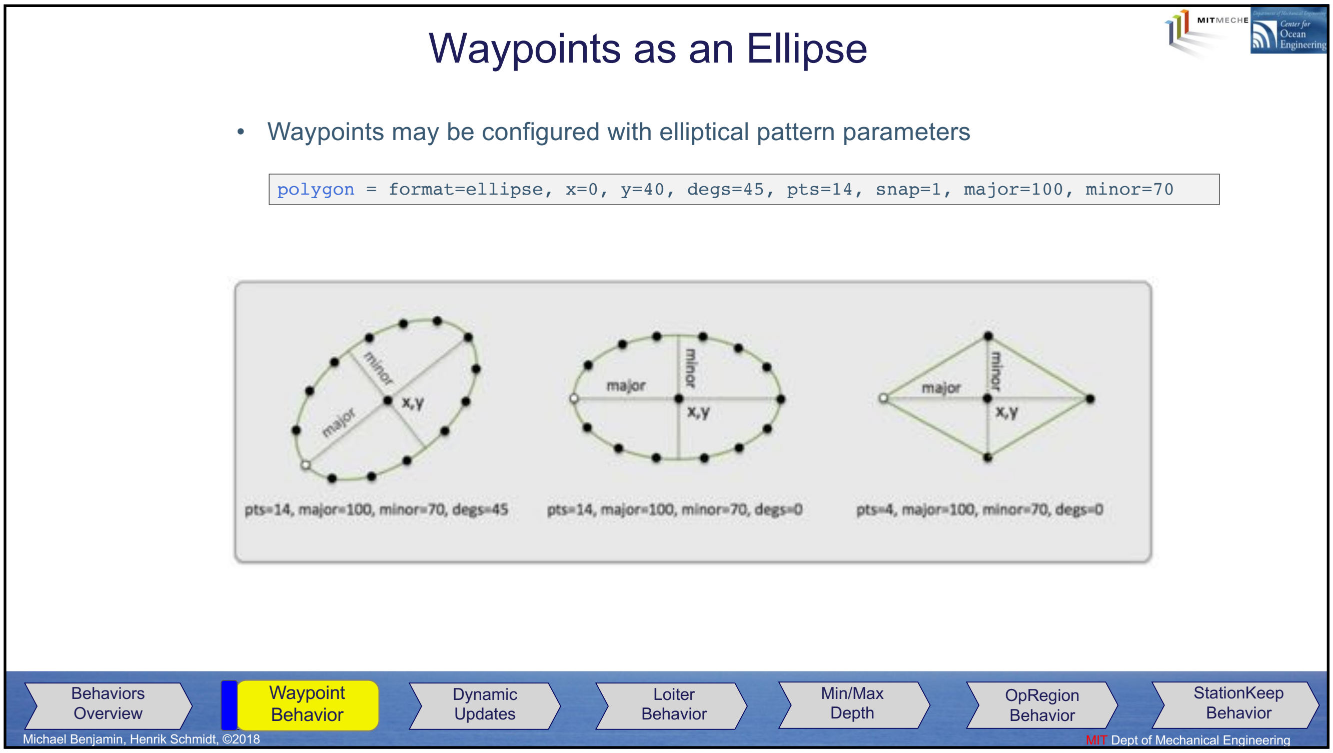


Waypoints as a Radial Polygon

- Waypoints may be configured with radial/circular pattern parameters

```
polygon = format=radial, x=0, y=40, radius=60, pts=6, snap=1
```





Waypoints as an Ellipse

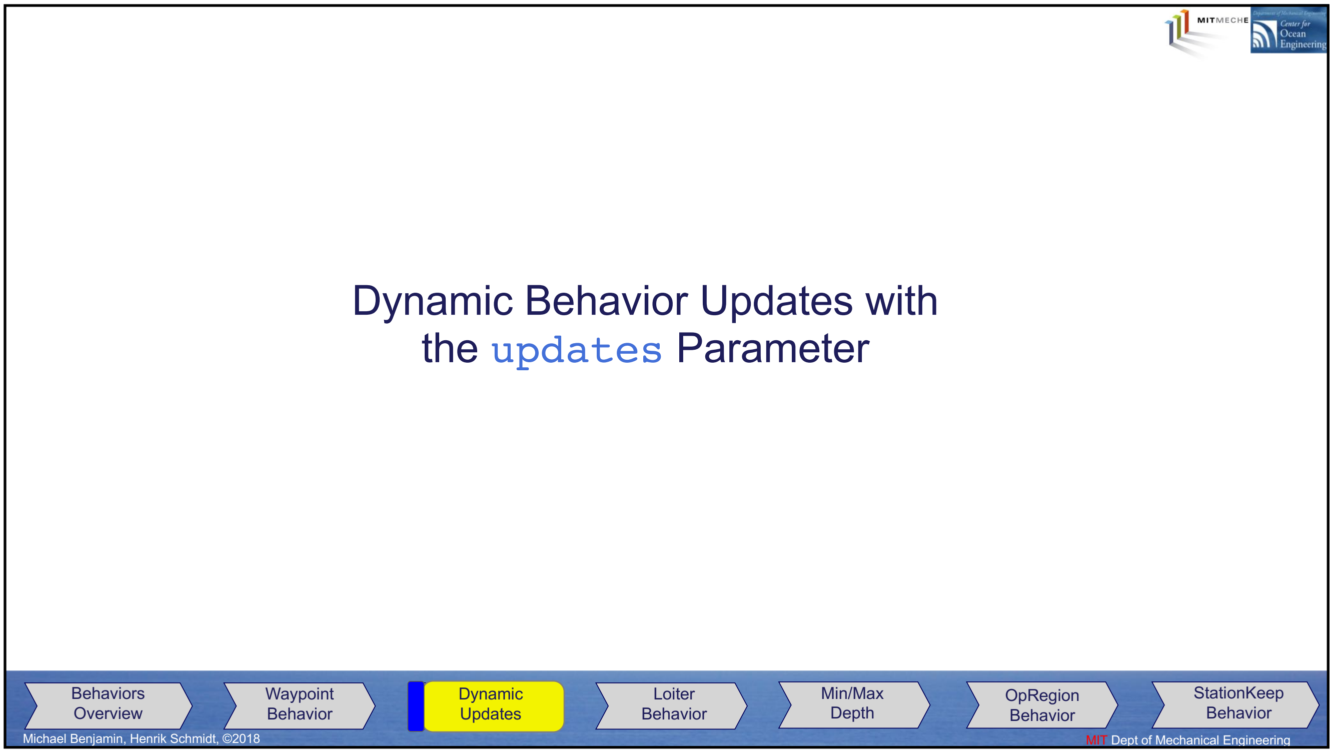
- Waypoints may be configured with elliptical pattern parameters

```
polygon = format=ellipse, x=0, y=40, degs=45, pts=14, snap=1, major=100, minor=70
```

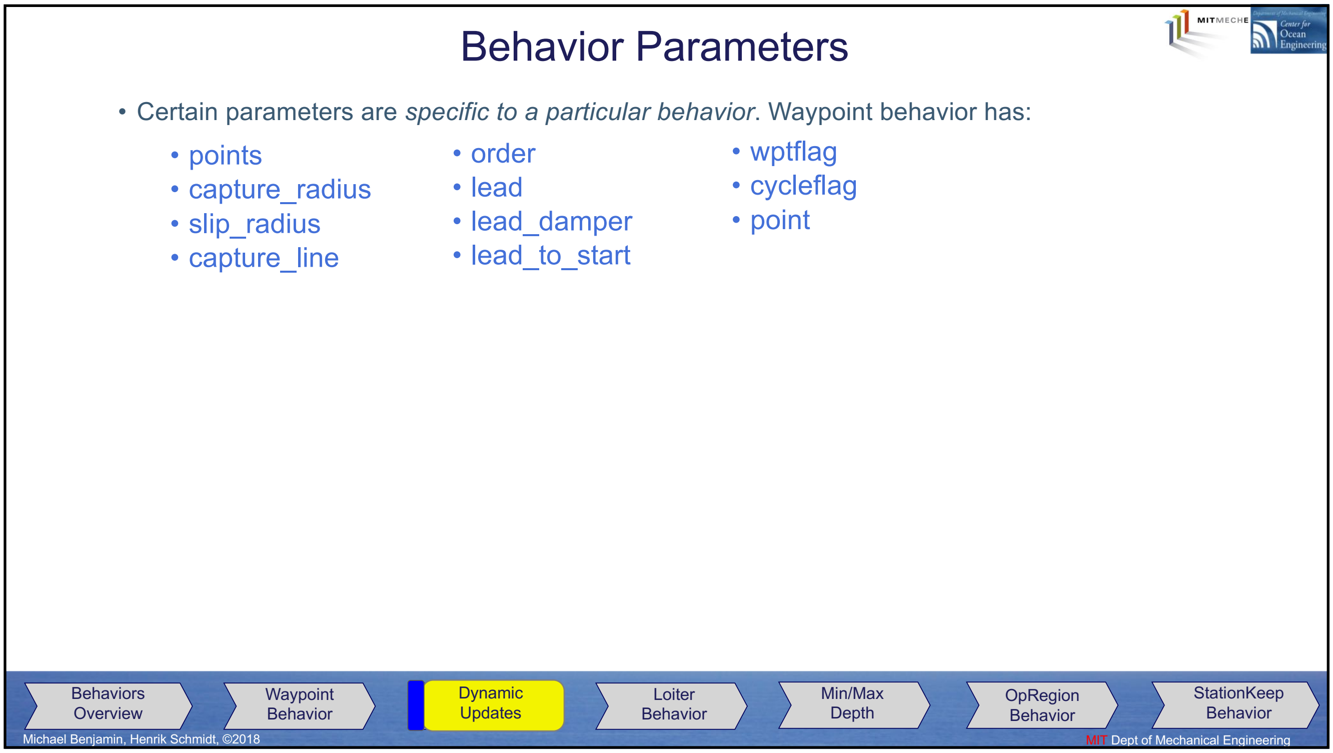
pts=14, major=100, minor=70, degs=45

pts=14, major=100, minor=70, degs=0

pts=4, major=100, minor=70, degs=0

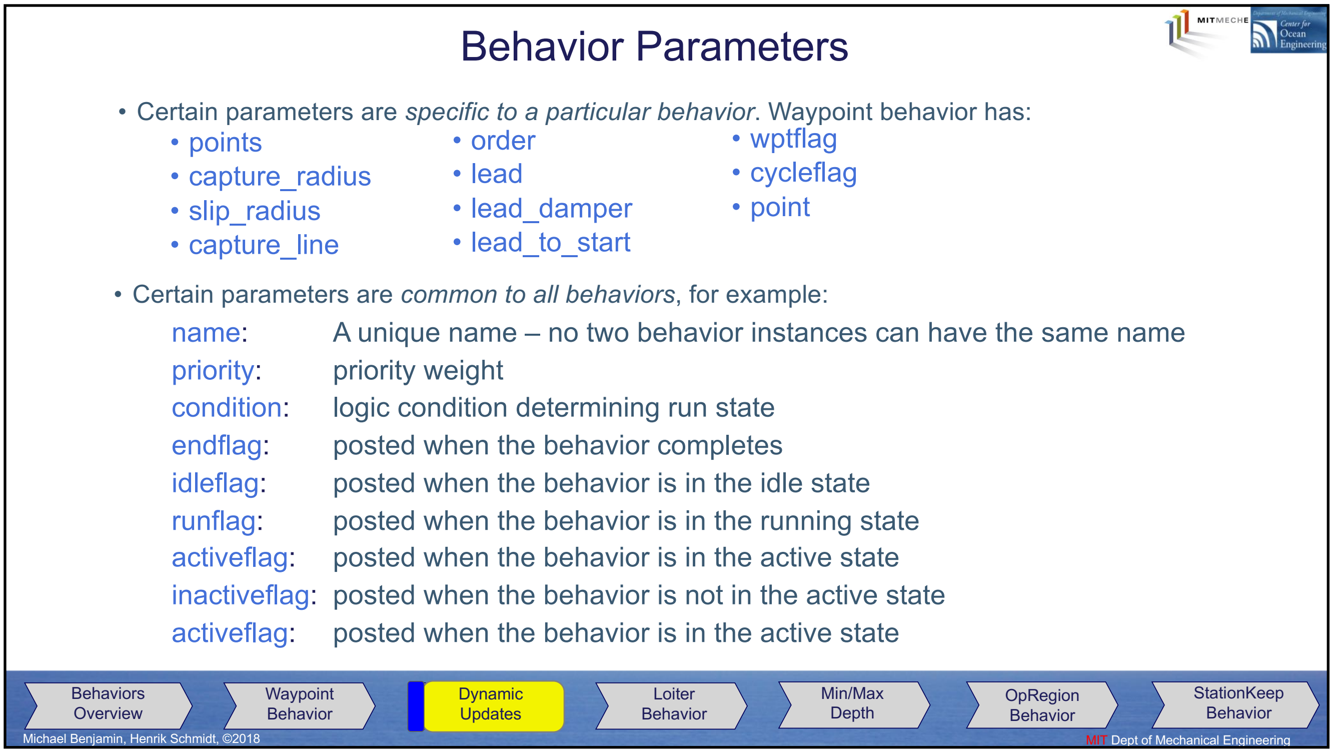


Dynamic Behavior Updates with the `updates` Parameter



Behavior Parameters

- Certain parameters are *specific to a particular behavior*. Waypoint behavior has:
  - points
  - capture_radius
  - slip_radius
  - capture_line
  - order
  - lead
  - lead_damper
  - lead_to_start
  - wptflag
  - cycleflag
  - point

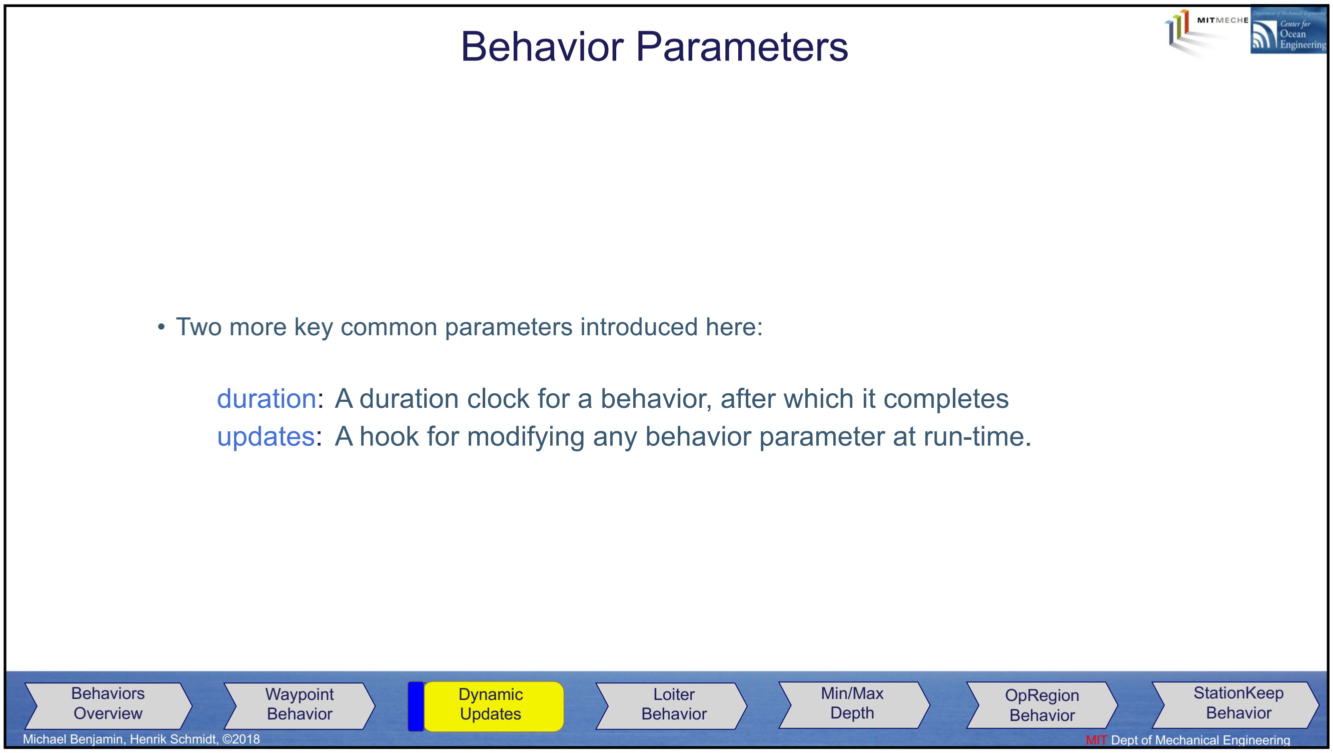


Behavior Parameters

- Certain parameters are *specific to a particular behavior*. Waypoint behavior has:
  - points
  - capture_radius
  - slip_radius
  - capture_line
  - order
  - lead
  - lead_damper
  - lead_to_start
  - wptflag
  - cycleflag
  - point

- Certain parameters are *common to all behaviors*, for example:

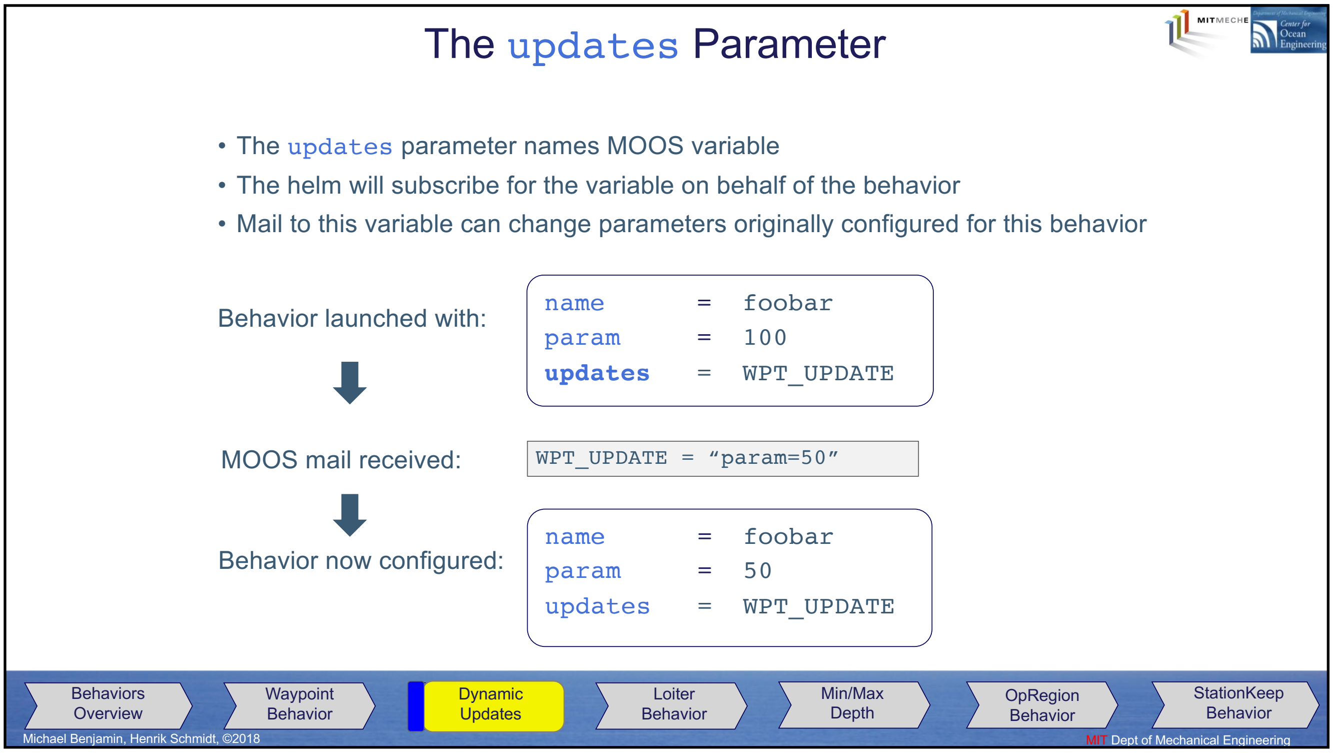
name:	A unique name – no two behavior instances can have the same name
priority:	priority weight
condition:	logic condition determining run state
endflag:	posted when the behavior completes
idleflag:	posted when the behavior is in the idle state
runflag:	posted when the behavior is in the running state
activeflag:	posted when the behavior is in the active state
inactiveflag:	posted when the behavior is not in the active state
activeflag:	posted when the behavior is in the active state



Behavior Parameters

- Two more key common parameters introduced here:

duration: A duration clock for a behavior, after which it completes
updates: A hook for modifying any behavior parameter at run-time.



The `updates` Parameter

- The `updates` parameter names MOOS variable
- The helm will subscribe for the variable on behalf of the behavior
- Mail to this variable can change parameters originally configured for this behavior

Behavior launched with:

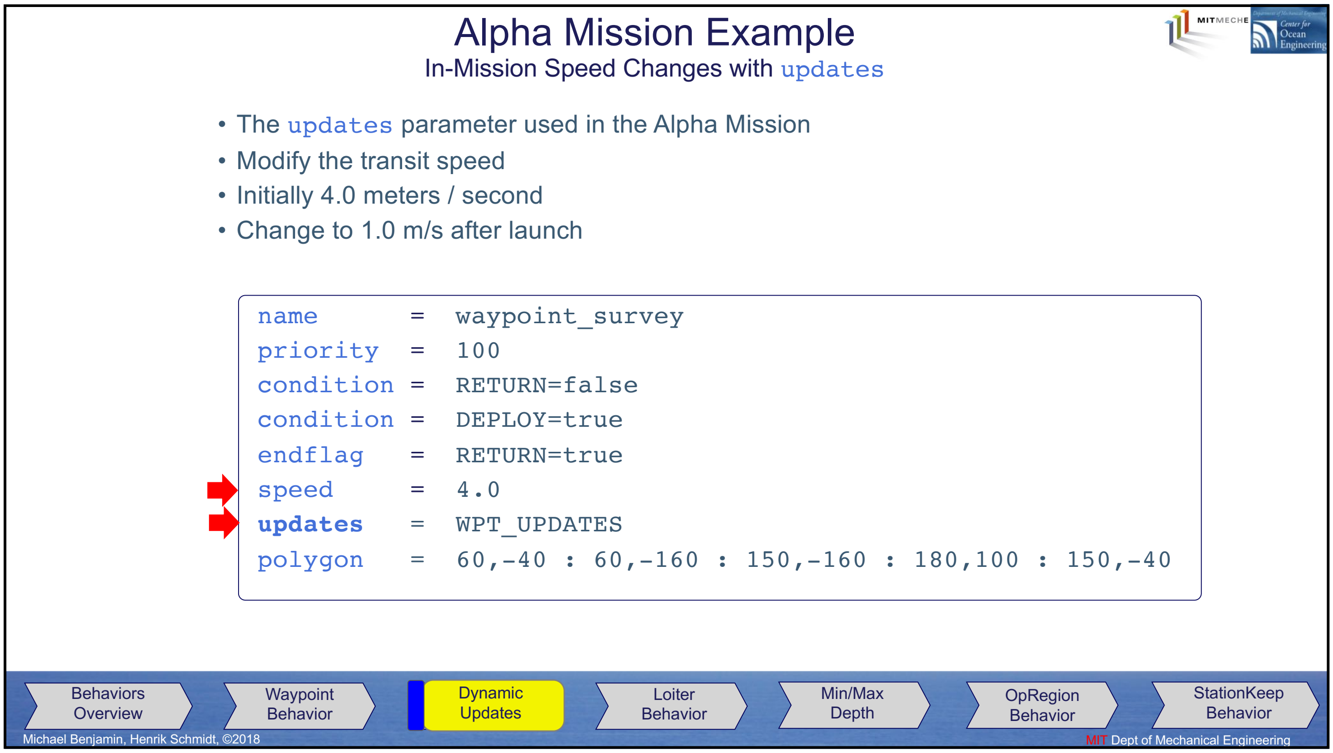
```
name      =  foobar
param     =  100
updates   =  WPT_UPDATE
```

MOOS mail received:

```
WPT_UPDATE = “param=50”
```

Behavior now configured:

```
name      =  foobar
param     =  50
updates   =  WPT_UPDATE
```



Alpha Mission Example

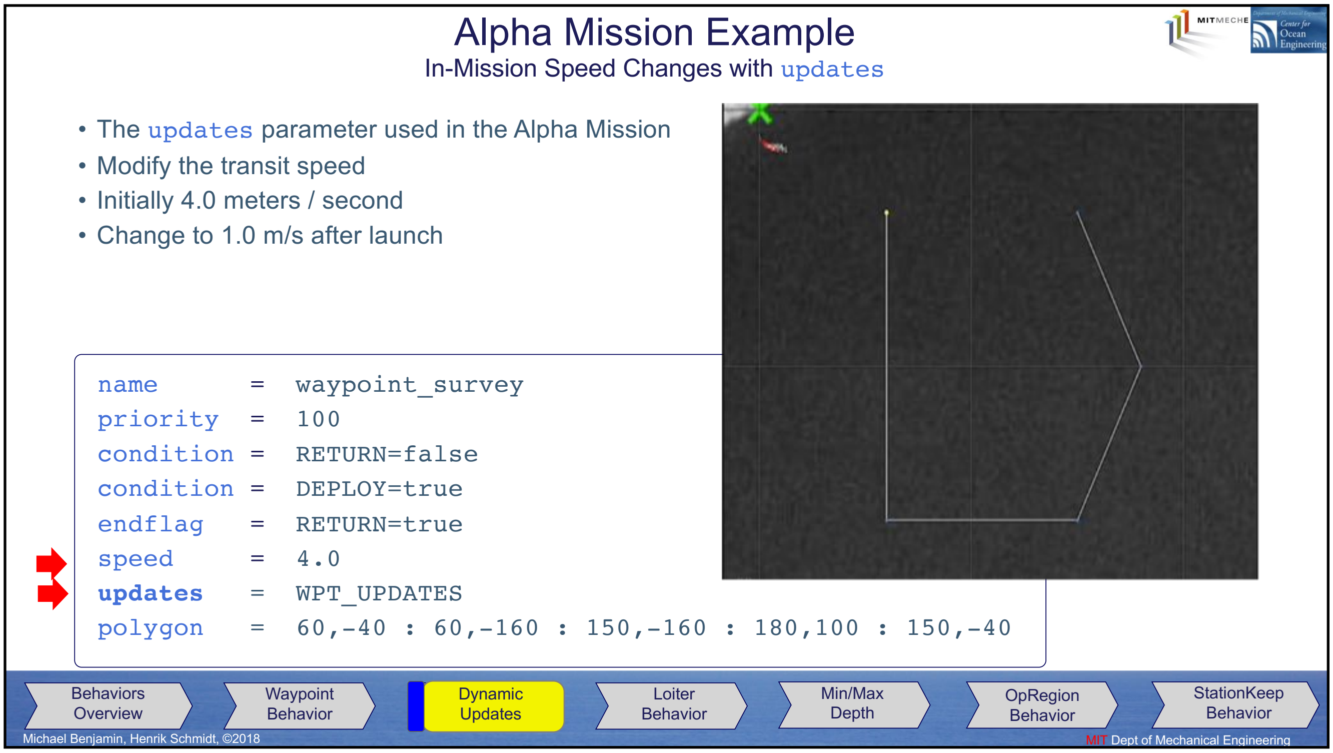
In-Mission Speed Changes with `updates`

- The `updates` parameter used in the Alpha Mission
- Modify the transit speed
- Initially 4.0 meters / second
- Change to 1.0 m/s after launch

```
name       =  waypoint_survey
priority   =  100
condition  =  RETURN=false
condition  =  DEPLOY=true
endflag    =  RETURN=true
speed      =  4.0
updates    =  WPT_UPDATES
polygon    =  60,-40 : 60,-160 : 150,-160 : 180,100 : 150,-40
```

Michael Benjamin, Henrik Schmidt, ©2018

MIT Dept of Mechanical Engineering



Alpha Mission Example

In-Mission Speed Changes with `updates`

- The `updates` parameter used in the Alpha Mission
- Modify the transit speed
- Initially 4.0 meters / second
- Change to 1.0 m/s after launch

```
name       =  waypoint_survey
priority   =  100
condition  =  RETURN=false
condition  =  DEPLOY=true
endflag    =  RETURN=true
speed      =  4.0
updates    =  WPT_UPDATES
polygon    =  60,-40 : 60,-160 : 150,-160 : 180,100 : 150,-40
```

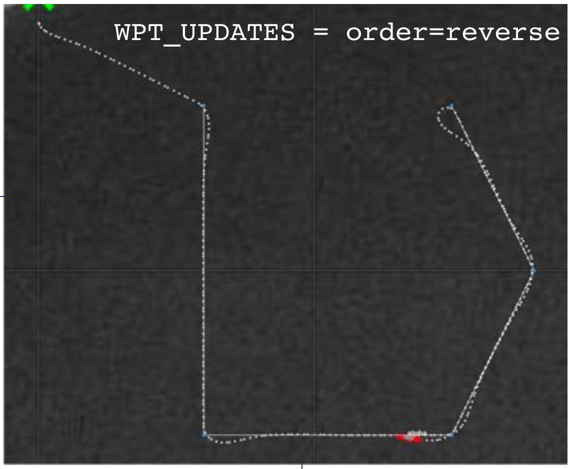
Michael Benjamin, Henrik Schmidt, ©2018

MIT Dept of Mechanical Engineering

Alpha Mission Example

In-Mission Reverse with `updates`

- After traversing the waypoints once, the `cycleflag` is published
- The `cycleflag` publishes to the `updates` variable, reversing the pattern direction for the second cycle.

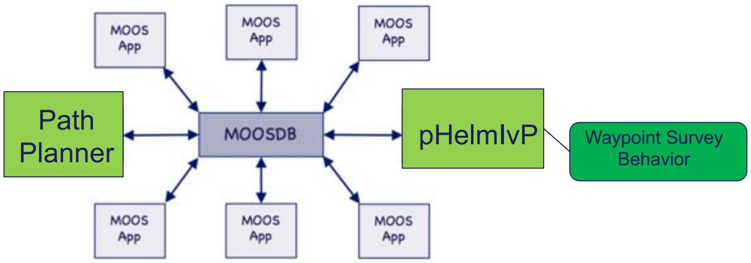


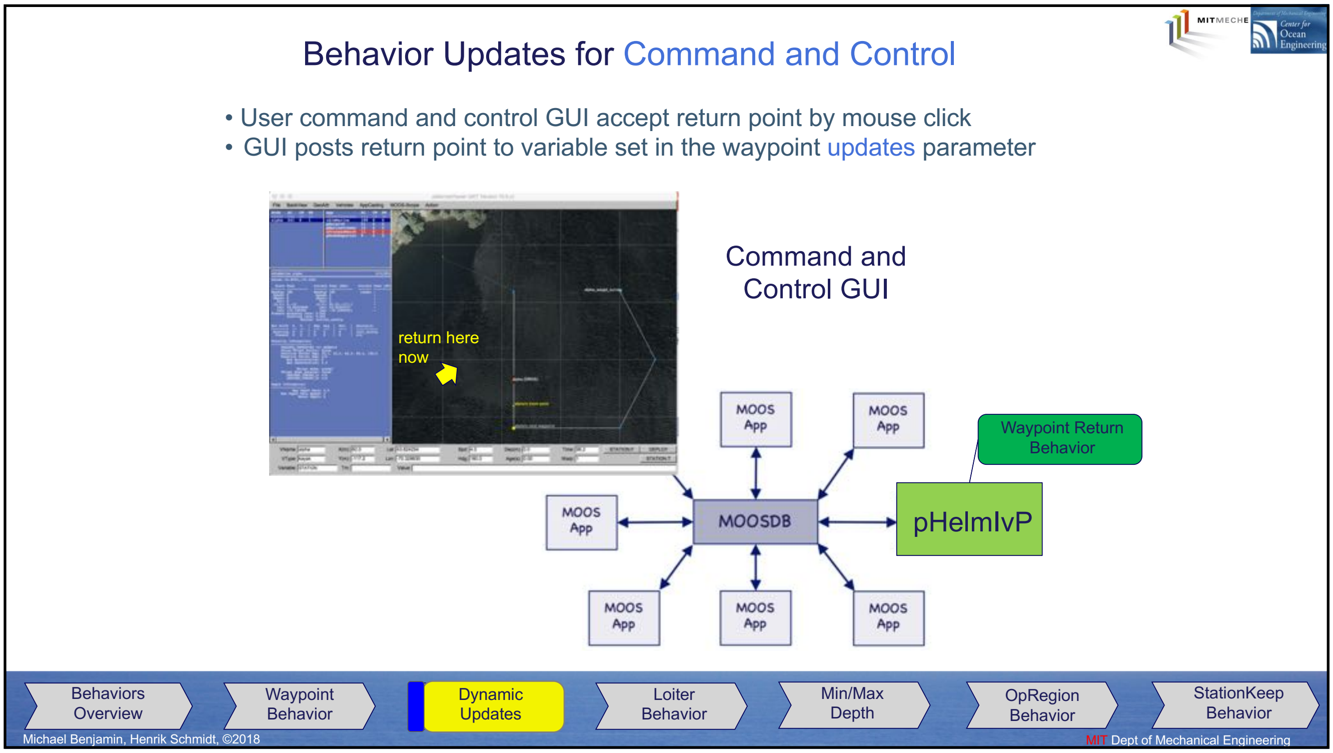
WPT_UPDATES = order=reverse

```
name       =  waypoint_survey
priority   =  100
condition  =  RETURN=false
condition  =  DEPLOY=true
endflag    =  RETURN=true
speed      =  4.0
cycleflag  =  WPT_UPDATES=order=reverse
updates    =  WPT_UPDATES
polygon    =  60,-40 : 60,-160 : 150,-160 : 180,100 : 150,-40
```

Behavior Updates for Path Planning

- Path planning MOOS App generates waypoints
- Behavior receives new waypoints through the updates





Behavior Updates for Command and Control

- User command and control GUI accept return point by mouse click
- GUI posts return point to variable set in the waypoint updates parameter

Command and Control GUI

MOOS App

MOOSDB

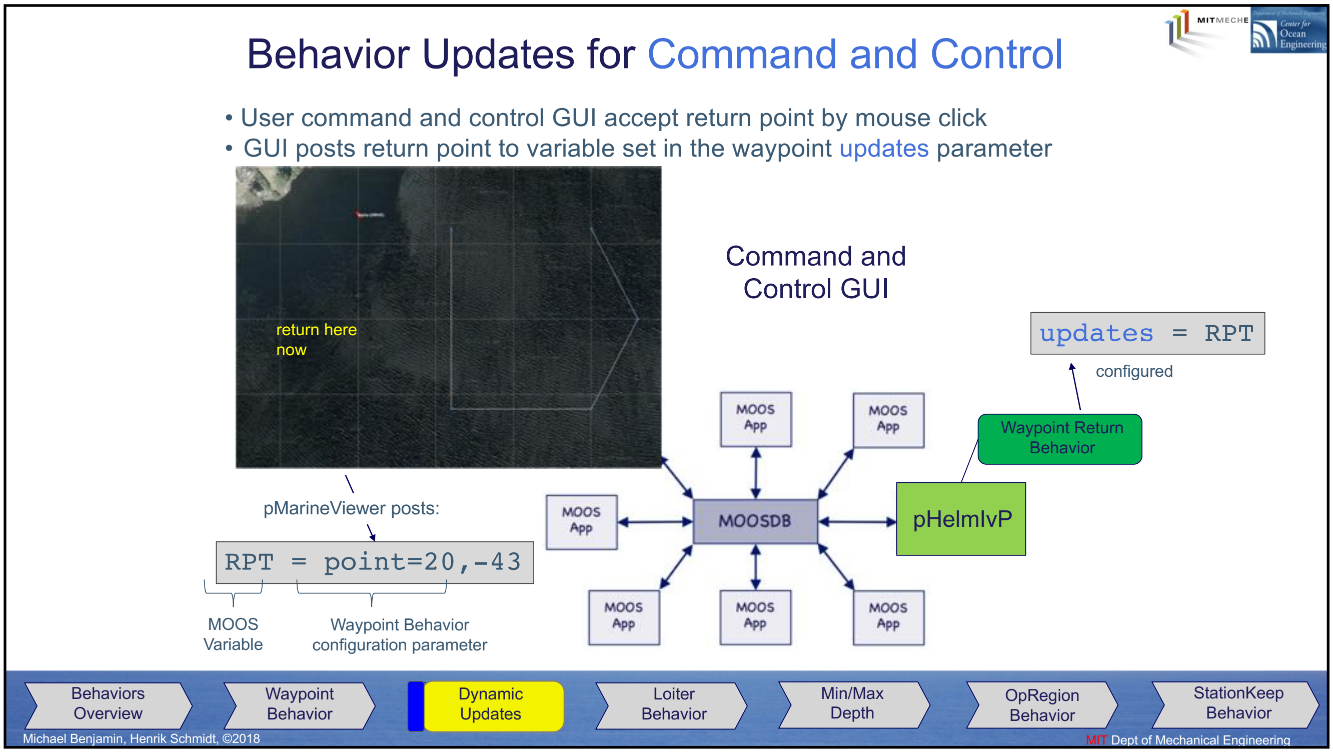
Waypoint Return Behavior

pHelmIvP

Behaviors Overview | Waypoint Behavior | Dynamic Updates | Loiter Behavior | Min/Max Depth | OpRegion Behavior | StationKeep Behavior

Michael Benjamin, Henrik Schmidt, ©2018

MIT Dept of Mechanical Engineering



Behavior Updates for Command and Control

- User command and control GUI accept return point by mouse click
- GUI posts return point to variable set in the waypoint updates parameter

Command and Control GUI

`updates = RPT`

configured

Waypoint Return Behavior

pHelmIvP

MOOSDB

MOOS App

pMarineViewer posts:

`RPT = point=20,-43`

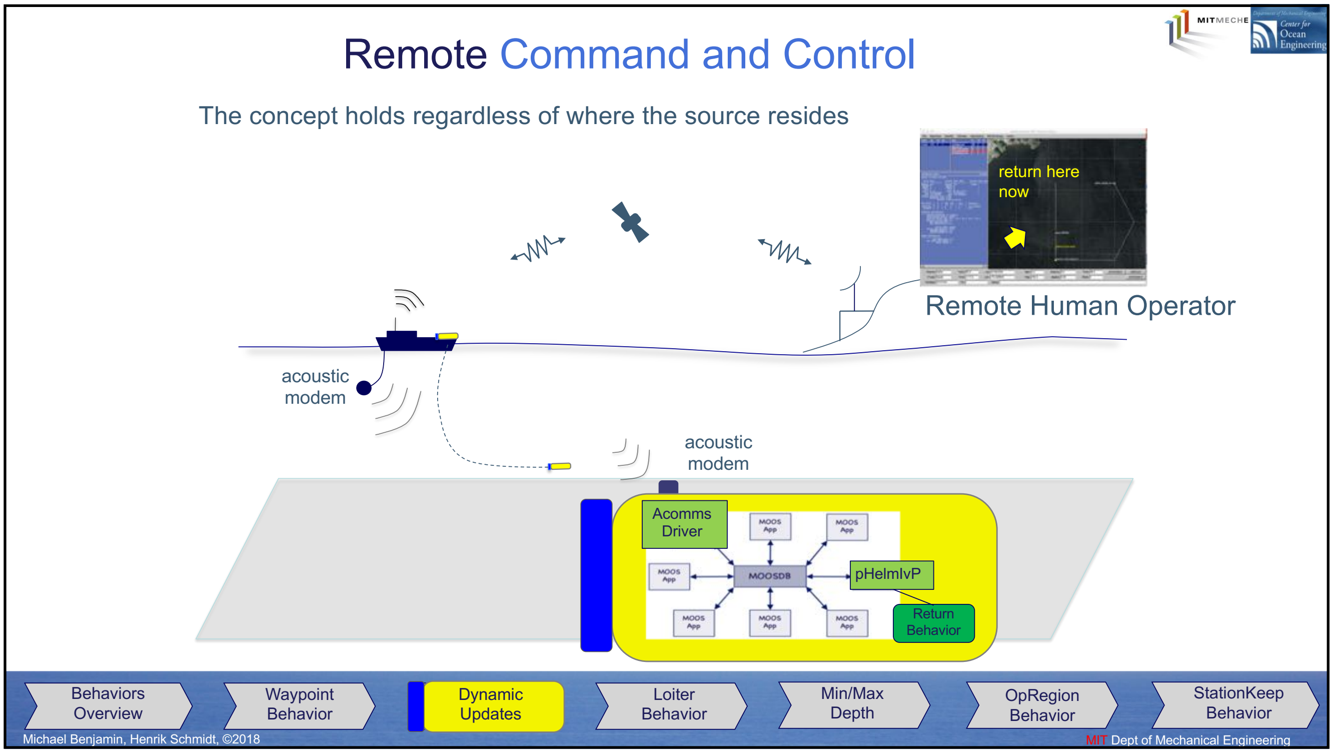
MOOS Variable

Waypoint Behavior configuration parameter

Behaviors Overview | Waypoint Behavior | Dynamic Updates | Loiter Behavior | Min/Max Depth | OpRegion Behavior | StationKeep Behavior

Michael Benjamin, Henrik Schmidt, ©2018

MIT Dept of Mechanical Engineering



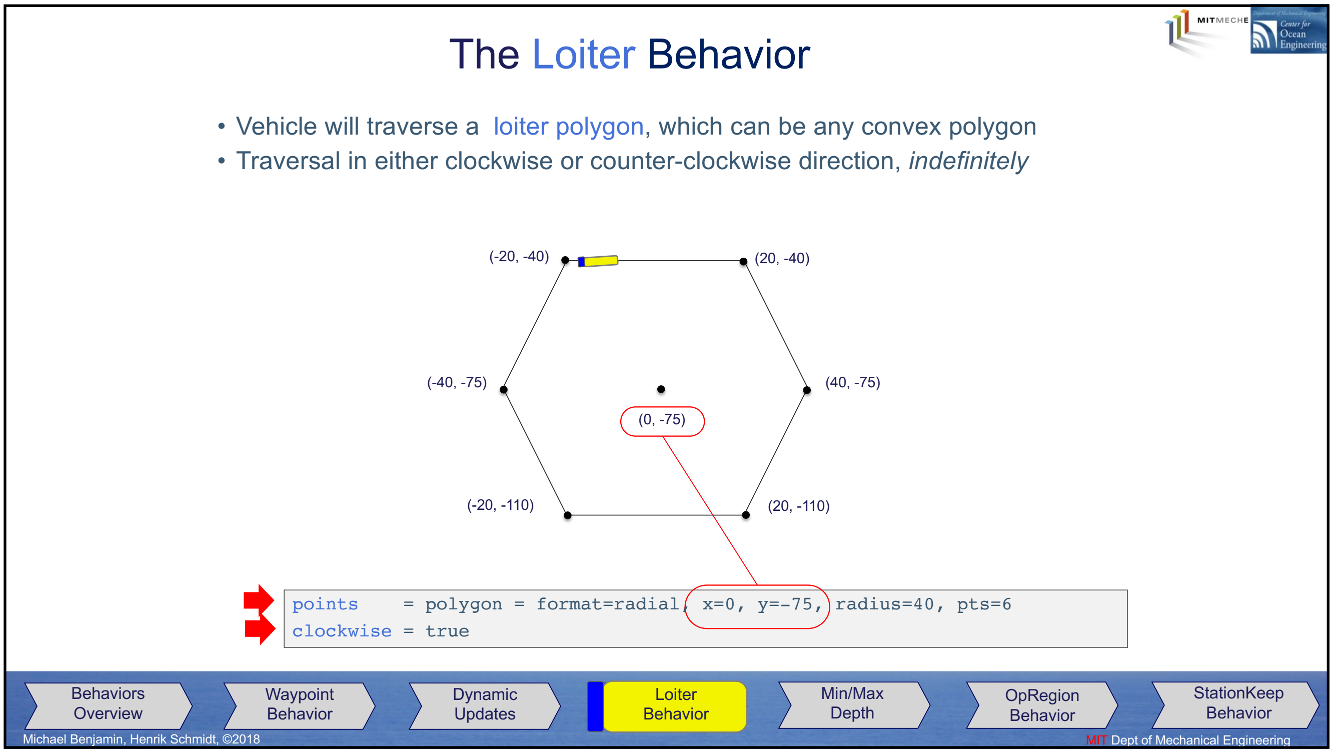
Remote Command and Control

The concept holds regardless of where the source resides

Behaviors Overview | Waypoint Behavior | Dynamic Updates | Loiter Behavior | Min/Max Depth | OpRegion Behavior | StationKeep Behavior

The Loiter Behavior

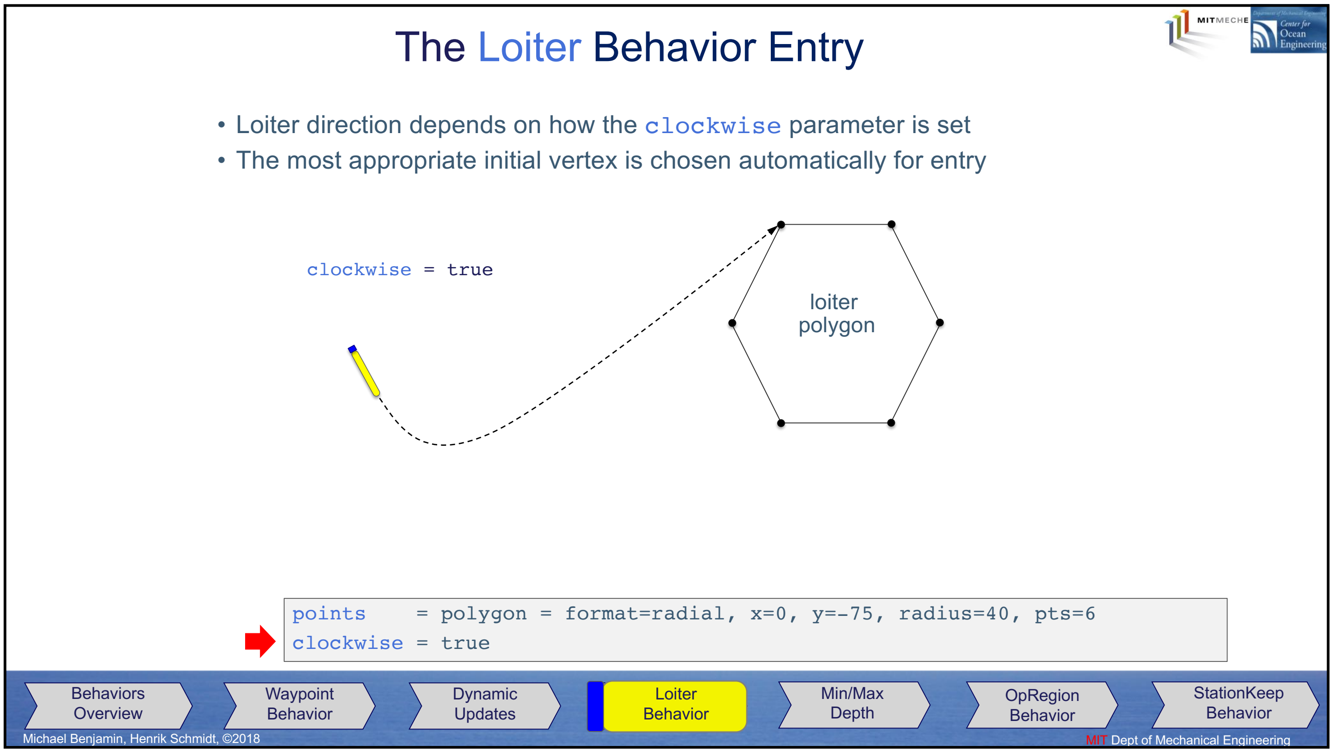
Behaviors Overview | Waypoint Behavior | Dynamic Updates | Loiter Behavior | Min/Max Depth | OpRegion Behavior | StationKeep Behavior



The Loiter Behavior

- Vehicle will traverse a loiter polygon, which can be any convex polygon
- Traversal in either clockwise or counter-clockwise direction, *indefinitely*

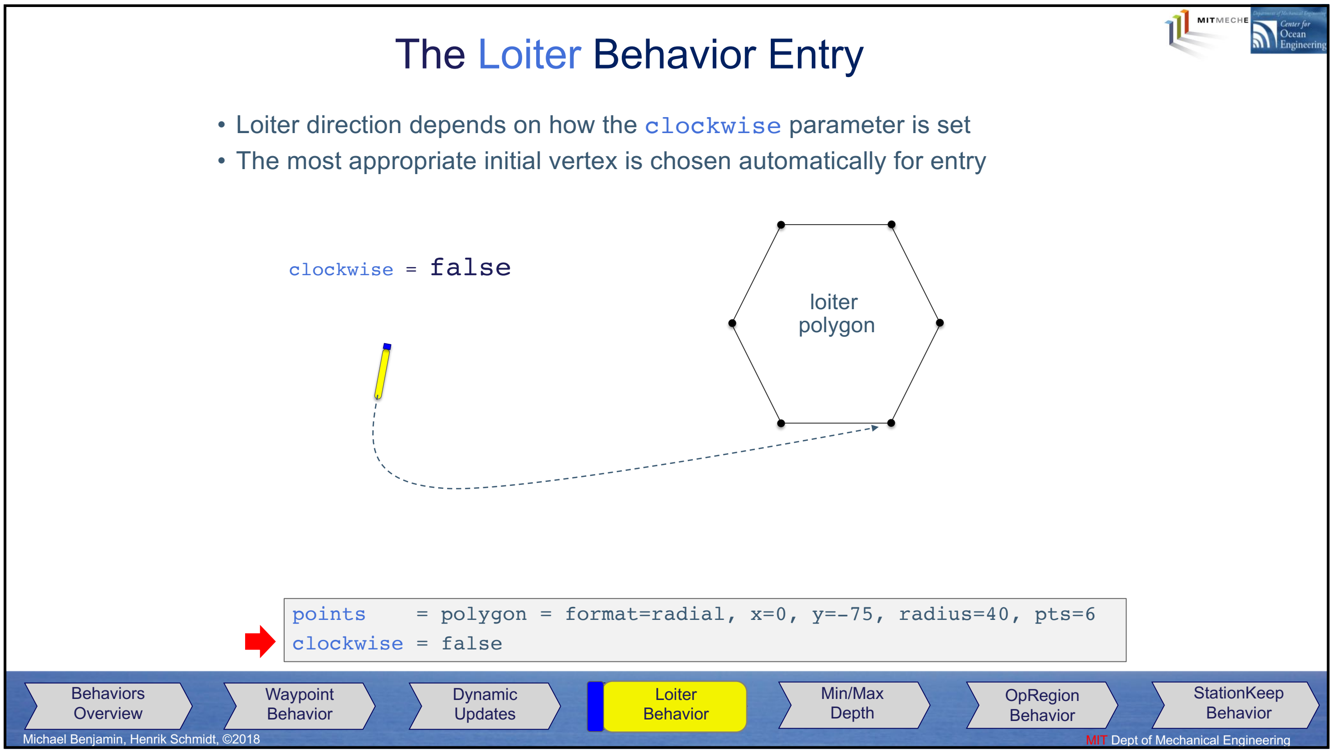
```
points    = polygon = format=radial, x=0, y=-75, radius=40, pts=6
clockwise = true
```



The Loiter Behavior Entry

- Loiter direction depends on how the `clockwise` parameter is set
- The most appropriate initial vertex is chosen automatically for entry

```
points    = polygon = format=radial, x=0, y=-75, radius=40, pts=6
clockwise = true
```



The Loiter Behavior Entry

- Loiter direction depends on how the `clockwise` parameter is set
- The most appropriate initial vertex is chosen automatically for entry

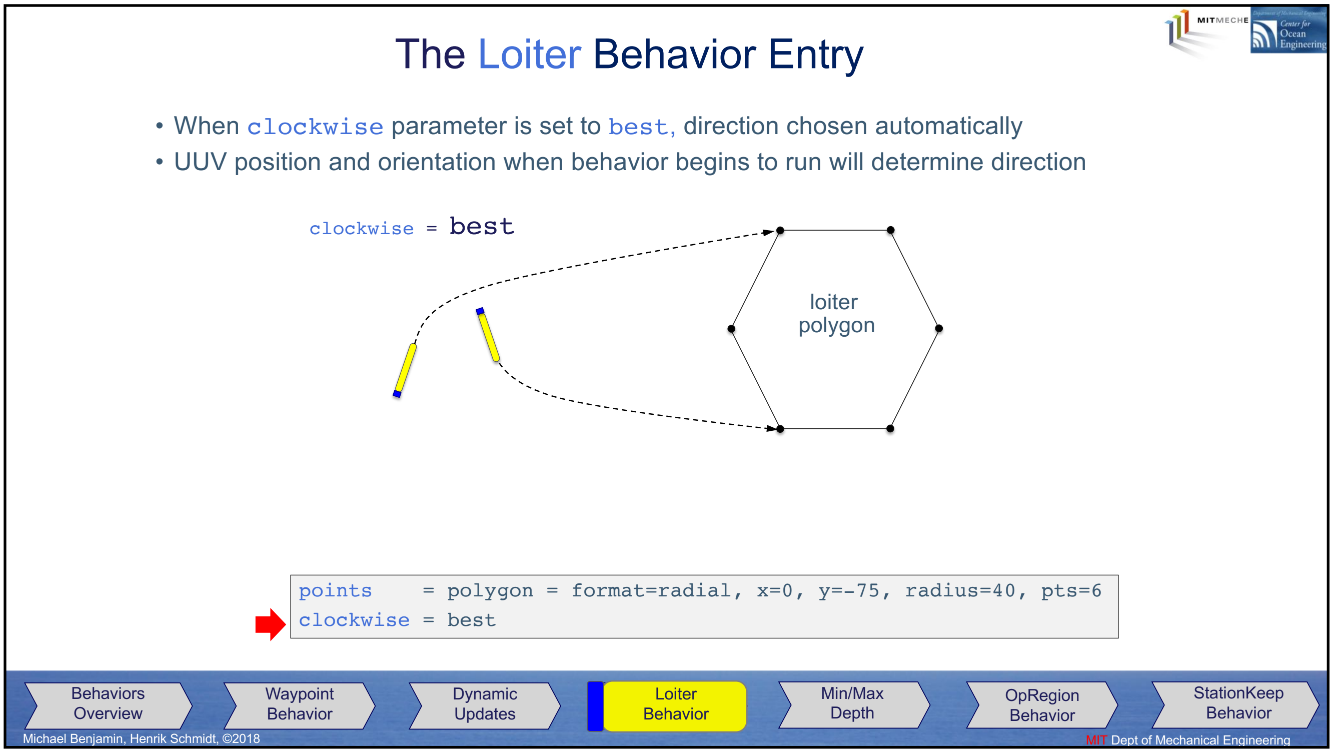
`clockwise = false`

```
points    = polygon = format=radial, x=0, y=-75, radius=40, pts=6
clockwise = false
```

Behaviors Overview | Waypoint Behavior | Dynamic Updates | Loiter Behavior | Min/Max Depth | OpRegion Behavior | StationKeep Behavior

Michael Benjamin, Henrik Schmidt, ©2018

MIT Dept of Mechanical Engineering



The Loiter Behavior Entry

- When `clockwise` parameter is set to `best`, direction chosen automatically
- UUV position and orientation when behavior begins to run will determine direction

`clockwise = best`

```
points    = polygon = format=radial, x=0, y=-75, radius=40, pts=6
clockwise = best
```

Behaviors Overview | Waypoint Behavior | Dynamic Updates | Loiter Behavior | Min/Max Depth | OpRegion Behavior | StationKeep Behavior

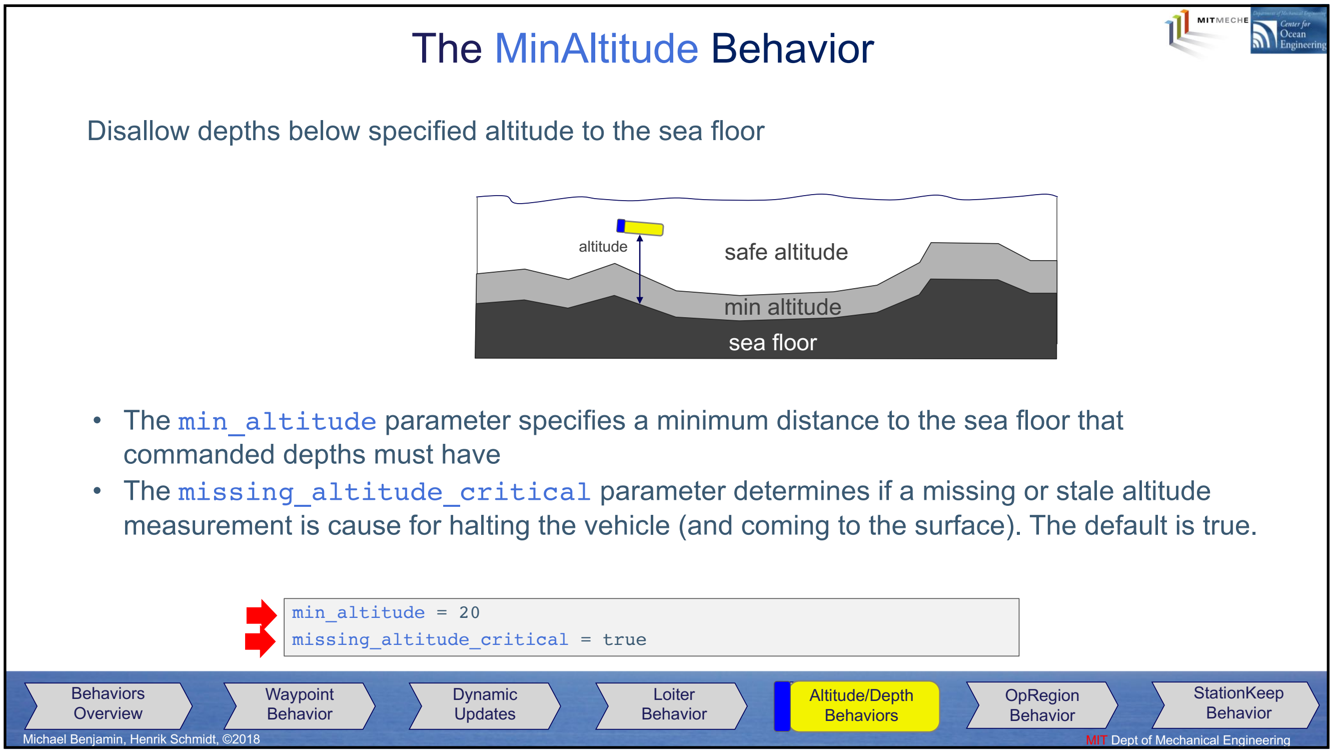
Michael Benjamin, Henrik Schmidt, ©2018

MIT Dept of Mechanical Engineering

Multi-Vehicle Loiter Example

- Note robustness on entry angle
- collision avoidance makes entry non-trivial

The MinAltitude and MaxDepth Behaviors

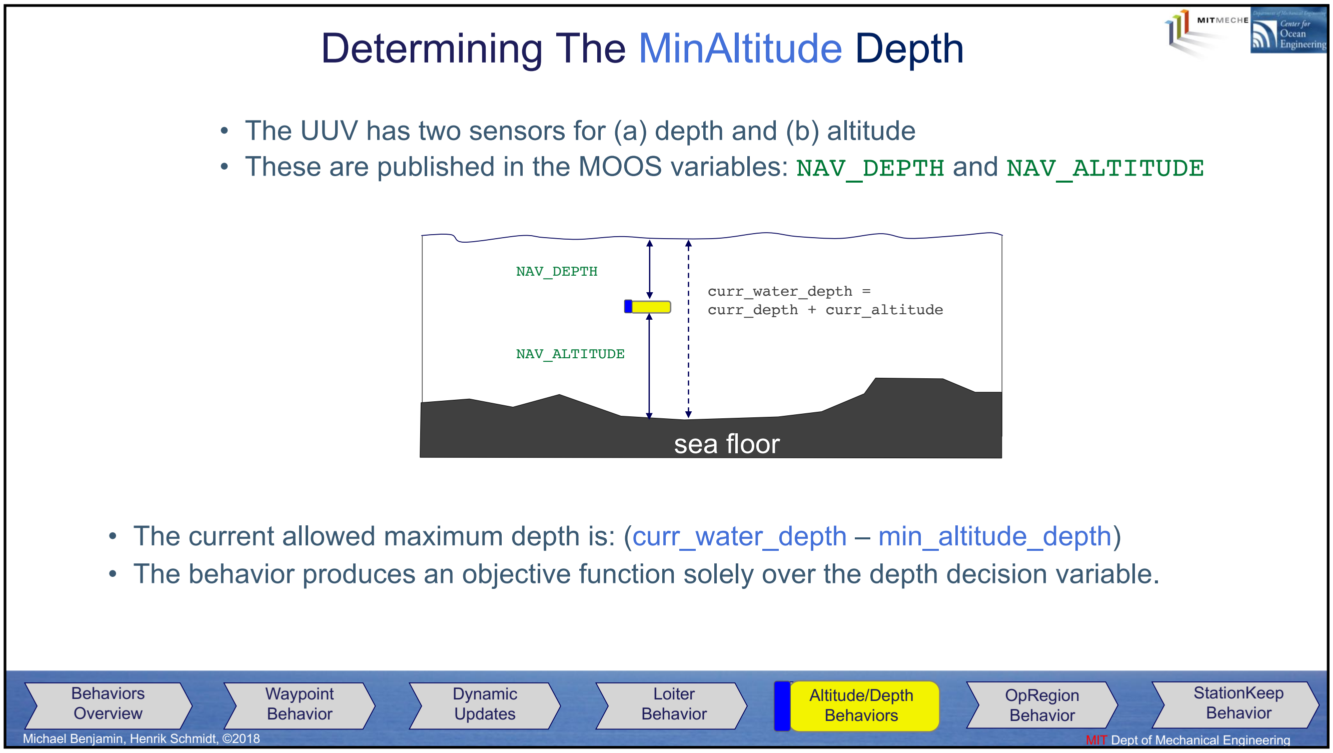


The MinAltitude Behavior

Disallow depths below specified altitude to the sea floor

- The `min_altitude` parameter specifies a minimum distance to the sea floor that commanded depths must have
- The `missing_altitude_critical` parameter determines if a missing or stale altitude measurement is cause for halting the vehicle (and coming to the surface). The default is true.

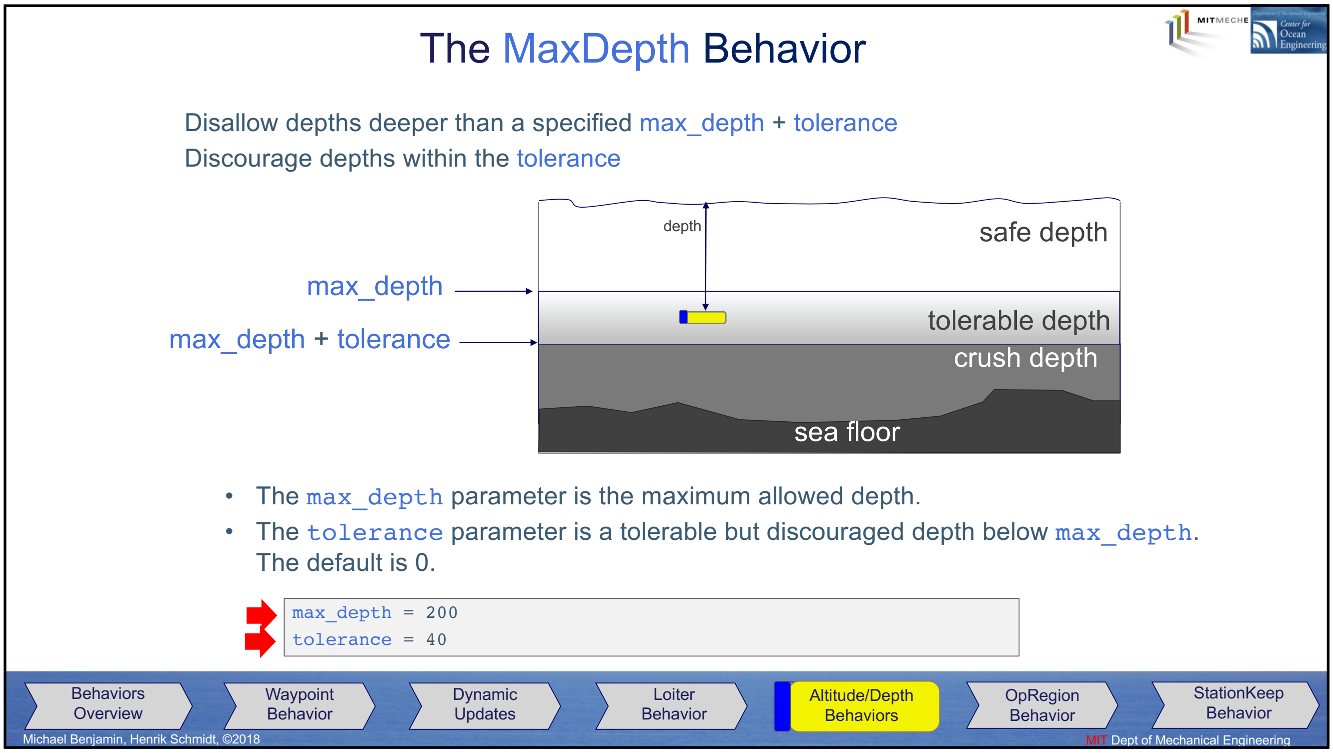
```
min_altitude = 20
missing_altitude_critical = true
```



Determining The MinAltitude Depth

- The UUV has two sensors for (a) depth and (b) altitude
- These are published in the MOOS variables: `NAV_DEPTH` and `NAV_ALTITUDE`

- The current allowed maximum depth is: (curr_water_depth – min_altitude_depth)
- The behavior produces an objective function solely over the depth decision variable.



The MaxDepth Behavior

Disallow depths deeper than a specified max_depth + tolerance
Discourage depths within the tolerance

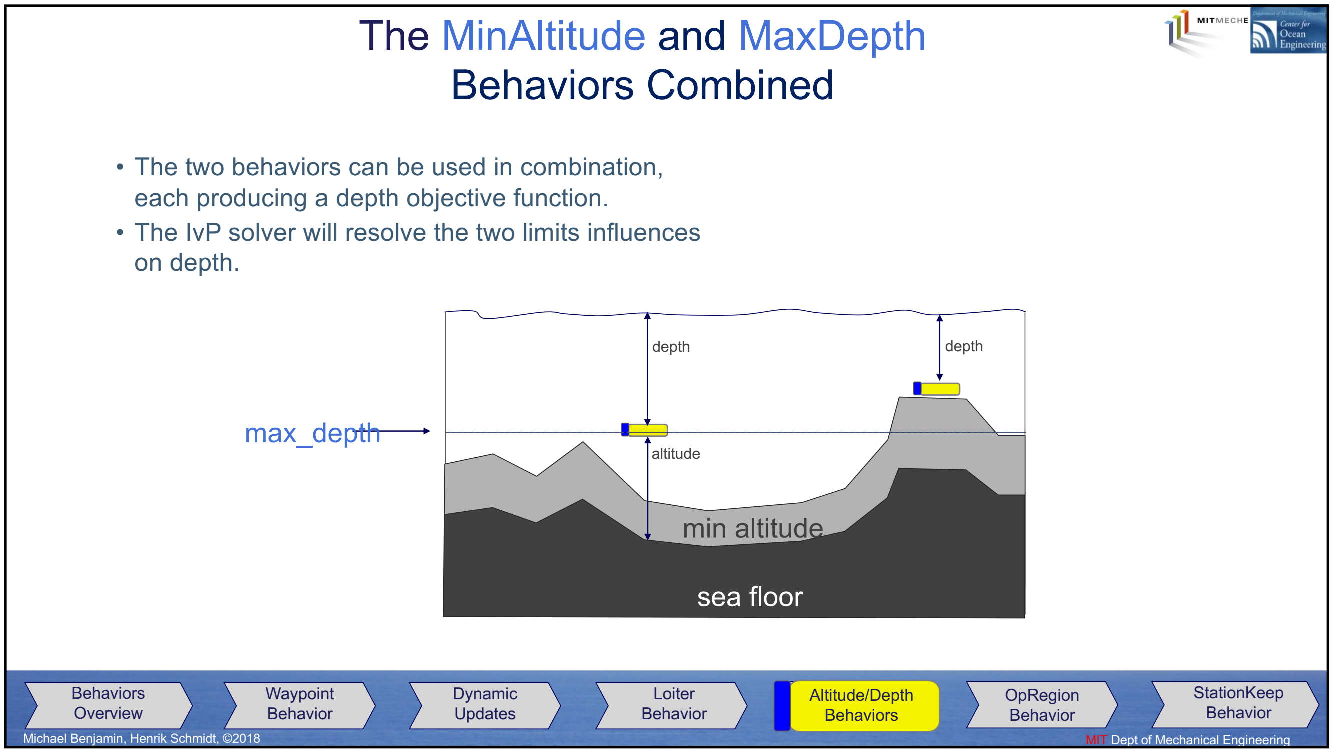
- The `max_depth` parameter is the maximum allowed depth.
- The `tolerance` parameter is a tolerable but discouraged depth below `max_depth`. The default is 0.

```
max_depth = 200
tolerance = 40
```

Behaviors Overview | Waypoint Behavior | Dynamic Updates | Loiter Behavior | Altitude/Depth Behaviors | OpRegion Behavior | StationKeep Behavior

Michael Benjamin, Henrik Schmidt, ©2018

MIT Dept of Mechanical Engineering



The MinAltitude and MaxDepth Behaviors Combined

- The two behaviors can be used in combination, each producing a depth objective function.
- The IvP solver will resolve the two limits influences on depth.

Behaviors Overview | Waypoint Behavior | Dynamic Updates | Loiter Behavior | Altitude/Depth Behaviors | OpRegion Behavior | StationKeep Behavior

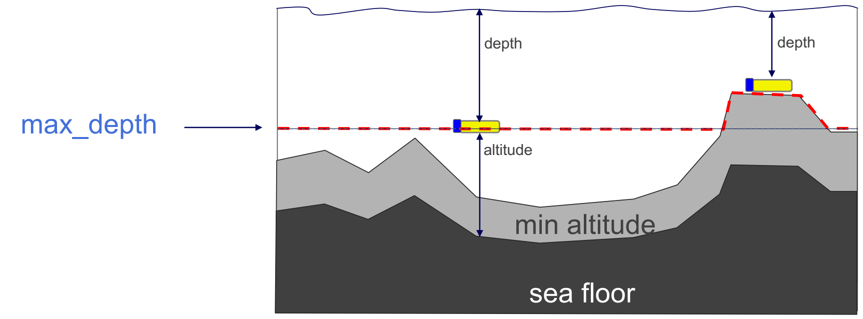
Michael Benjamin, Henrik Schmidt, ©2018

MIT Dept of Mechanical Engineering

The MinAltitude and MaxDepth Behaviors Combined

- The two behaviors can be used in combination, each producing a depth objective function.
- The IvP solver will resolve the two limits influences on depth.

Combined allowed depth



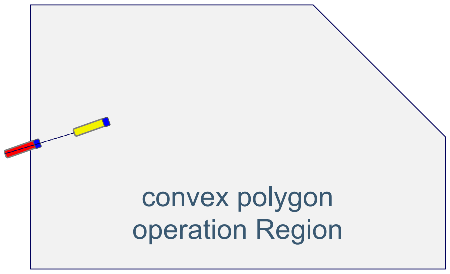
The OpRegion Behavior

The OpRegion Behavior

OpRegion behavior provides four different types of safety functionality:

- a boundary box given by a convex polygon in the x-y or lat-lon plane
- an overall timeout
- a depth limit
- an altitude limit

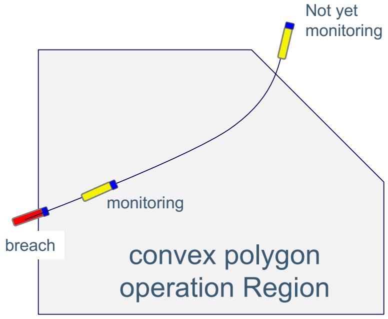
  - The behavior does not produce an objective function to influence the vehicle to avoid violating these safety constraints.
  - This behavior merely monitors the constraints and posts an error which results in the posting of all-stop commands,



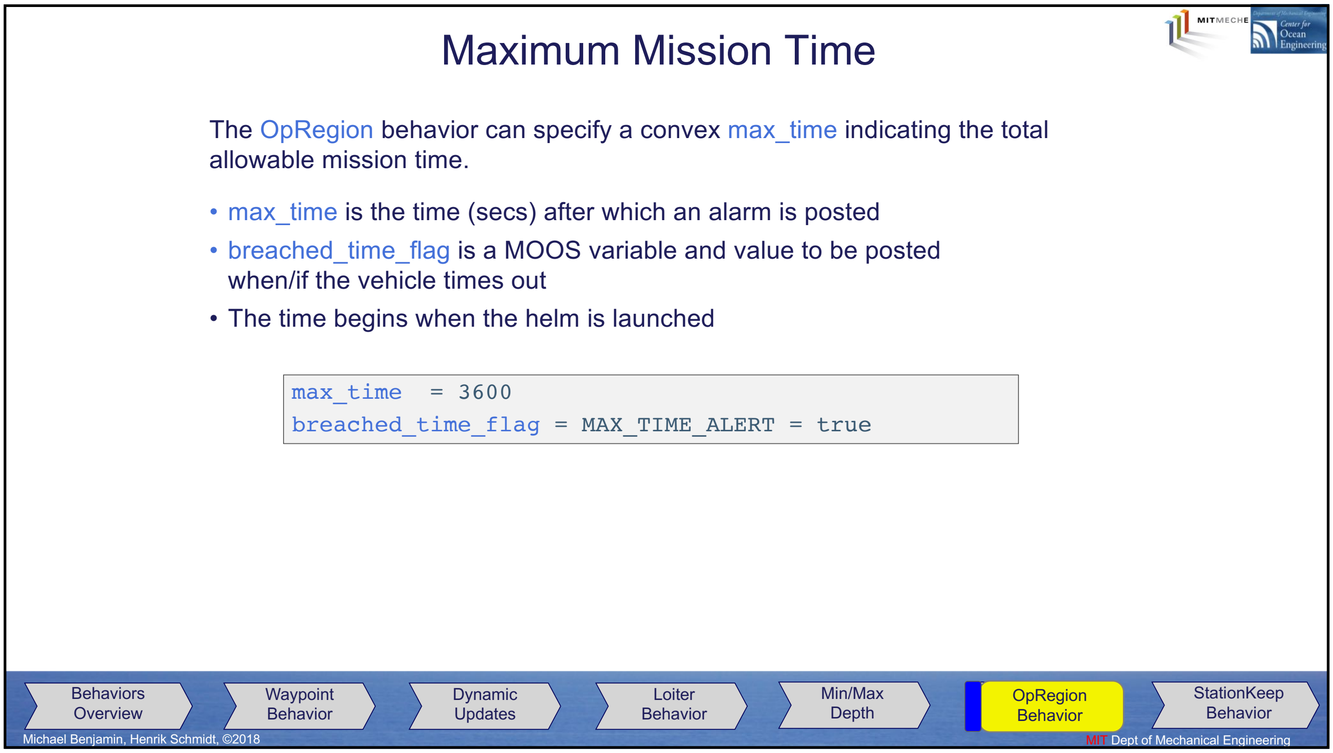
Polygon Containment

- The OpRegion behavior can specify a convex polygon indicating the allowable area of operation for the vehicle

  - Monitoring is not active until the vehicle enters the polygon
  - trigger_entry_time is the time (secs) within the polygon before monitoring becomes active
  - trigger_exit_time is the time (secs) outside the polygon before alarm is triggered
  - breached_poly_flag is a MOOS variable and value to be posted when/if the vehicle exits the polygon region.



```
polygon = 0,-50:0,-150:150,-150:150,-50
trigger_entry_time = 1
trigger_exit_time  = 1
breached_poly_flag = COME_TO_SURFACE = true
```



Maximum Mission Time

The OpRegion behavior can specify a convex max_time indicating the total allowable mission time.

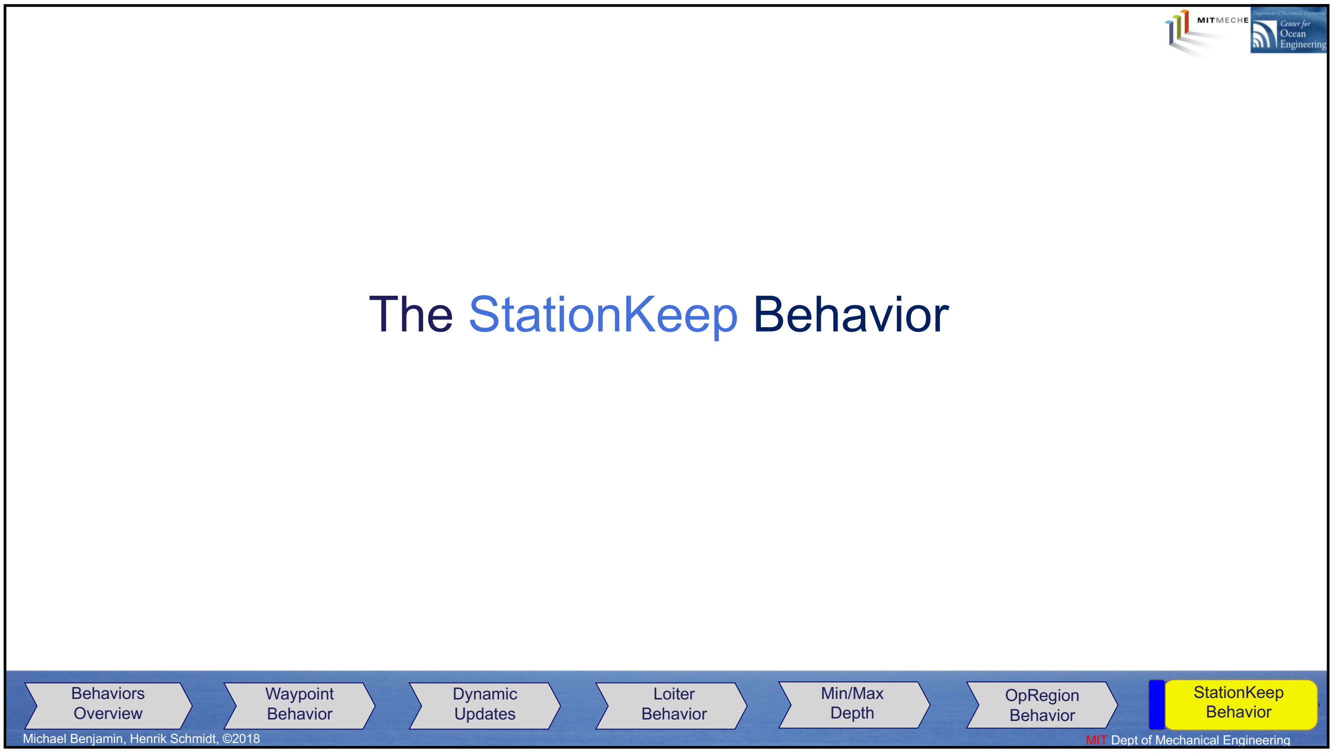
- max_time is the time (secs) after which an alarm is posted
- breached_time_flag is a MOOS variable and value to be posted when/if the vehicle times out
- The time begins when the helm is launched

```
max_time   = 3600
breached_time_flag = MAX_TIME_ALERT = true
```

Behaviors Overview
Waypoint Behavior
Dynamic Updates
Loiter Behavior
Min/Max Depth
OpRegion Behavior
StationKeep Behavior

Michael Benjamin, Henrik Schmidt, ©2018

MIT Dept of Mechanical Engineering

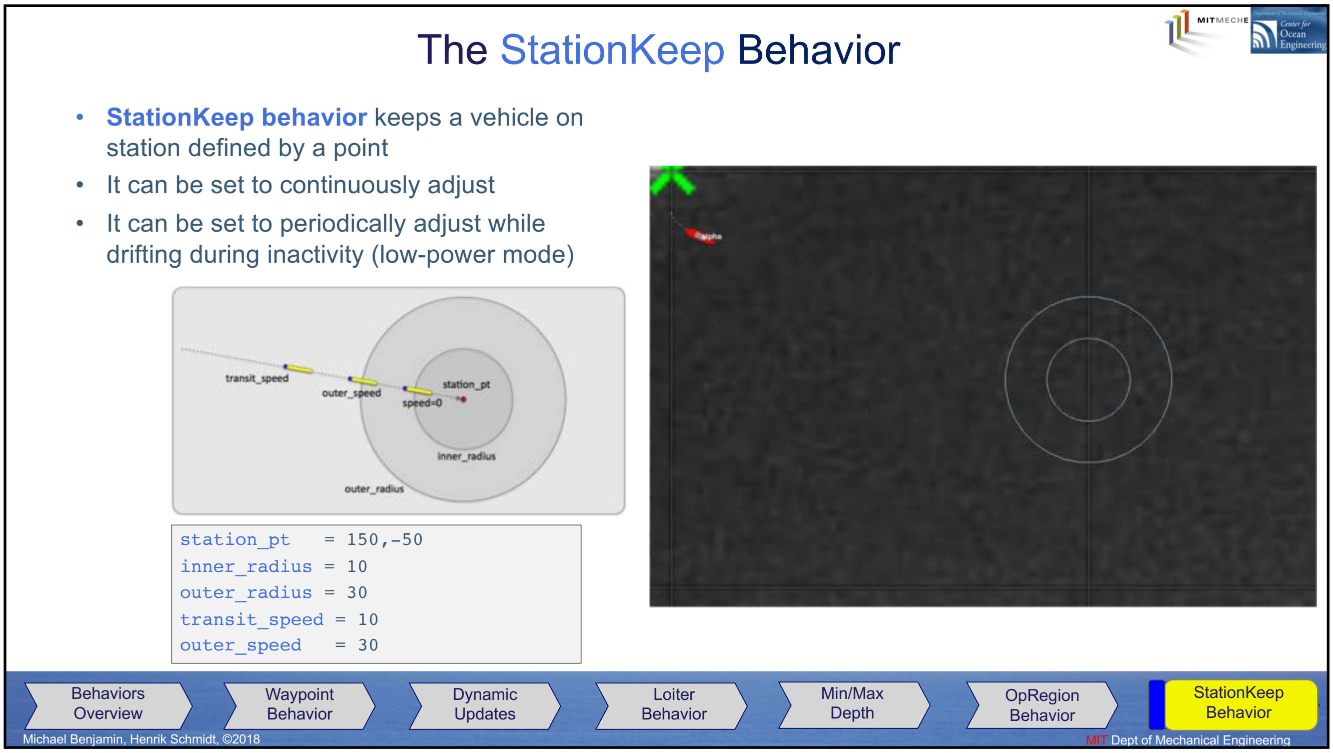


The StationKeep Behavior

Behaviors Overview
Waypoint Behavior
Dynamic Updates
Loiter Behavior
Min/Max Depth
OpRegion Behavior
StationKeep Behavior

Michael Benjamin, Henrik Schmidt, ©2018

MIT Dept of Mechanical Engineering



The StationKeep Behavior

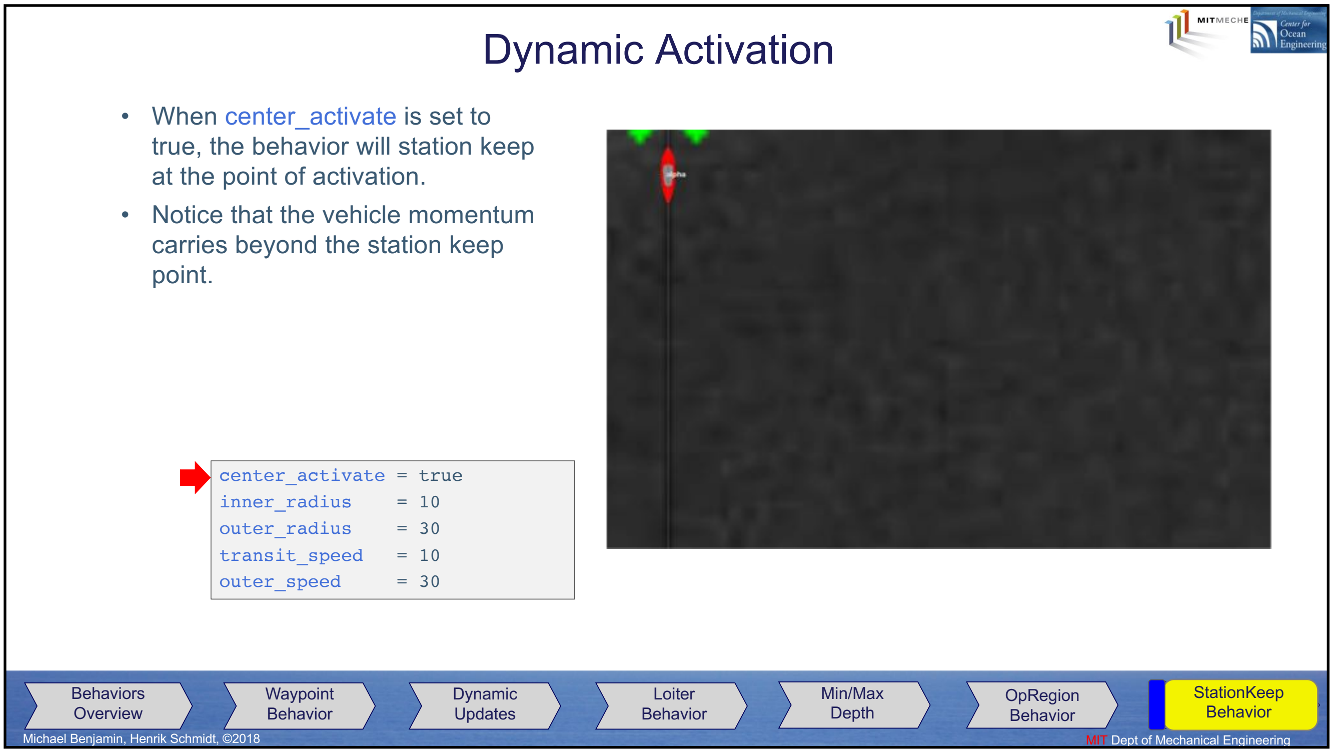
- **StationKeep behavior** keeps a vehicle on station defined by a point
- It can be set to continuously adjust
- It can be set to periodically adjust while drifting during inactivity (low-power mode)

```
station_pt   = 150,-50
inner_radius = 10
outer_radius = 30
transit_speed = 10
outer_speed   = 30
```

Behaviors Overview | Waypoint Behavior | Dynamic Updates | Loiter Behavior | Min/Max Depth | OpRegion Behavior | StationKeep Behavior

Michael Benjamin, Henrik Schmidt, ©2018

MIT Dept of Mechanical Engineering



Dynamic Activation

- When center_activate is set to true, the behavior will station keep at the point of activation.
- Notice that the vehicle momentum carries beyond the station keep point.

```
center_activate  = true
inner_radius     = 10
outer_radius     = 30
transit_speed    = 10
outer_speed      = 30
```

Behaviors Overview | Waypoint Behavior | Dynamic Updates | Loiter Behavior | Min/Max Depth | OpRegion Behavior | StationKeep Behavior

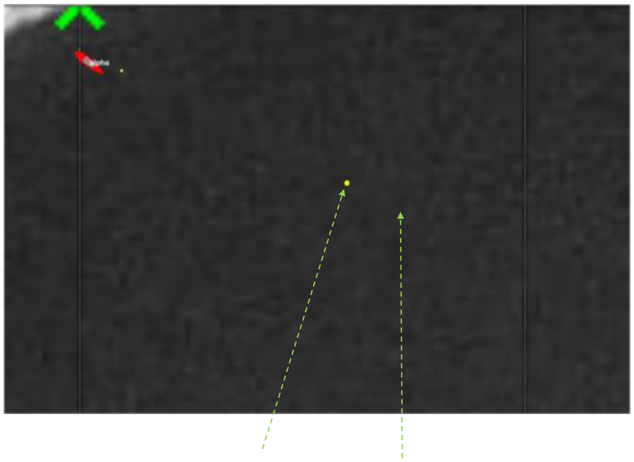
Michael Benjamin, Henrik Schmidt, ©2018

MIT Dept of Mechanical Engineering

Dynamic Activation

- When center_activate is set to true, the behavior will station keep at the point of activation.
- Notice that the vehicle momentum carries beyond the station keep point
- The swing_time parameter is the number of seconds after activation that the station point is marked

```
center_activate = true
swing time      = 10
inner_radius    = 10
outer_radius    = 30
transit_speed   = 10
outer_speed     = 30
```



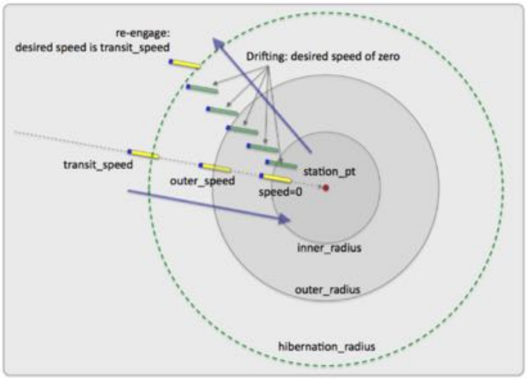
point of activation

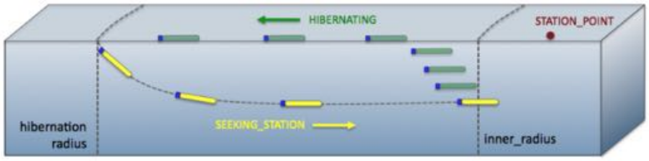
actual station point

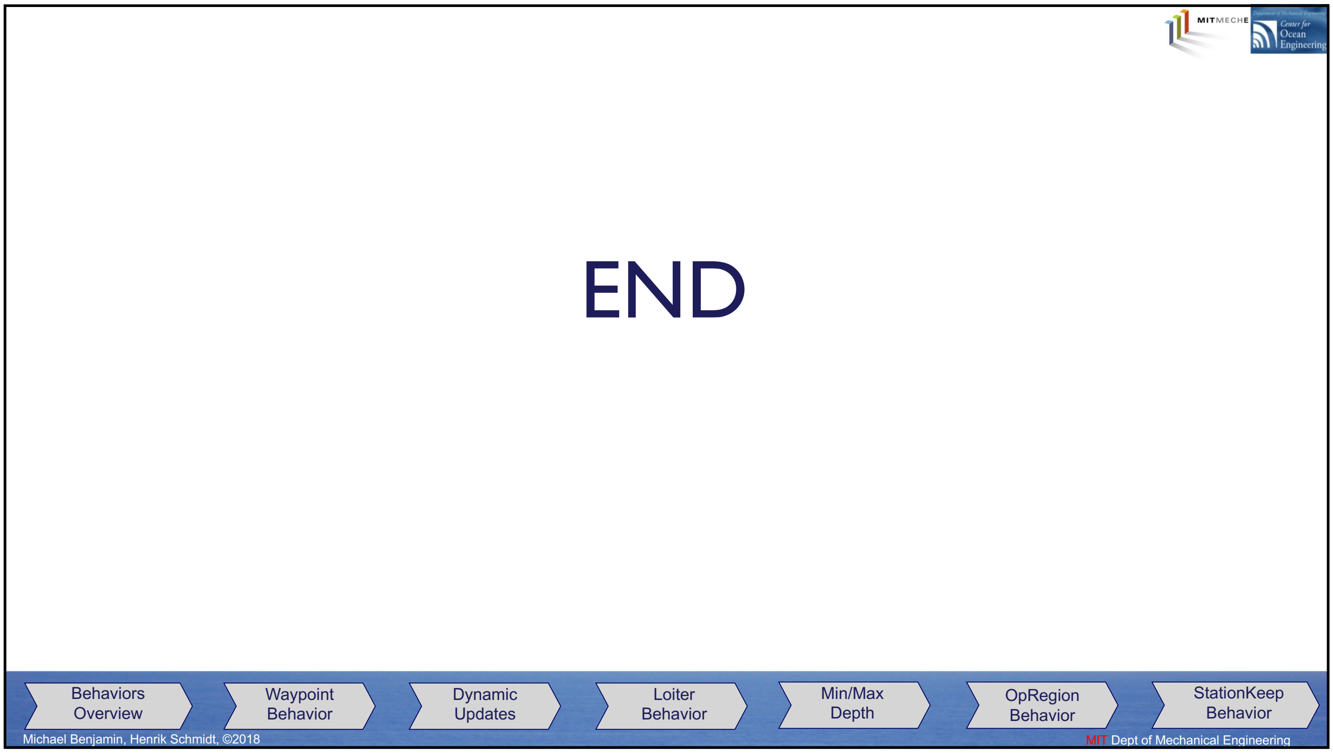
Low Power Station Keeping

- The hybernation_radius is a distance within which no corrective position keeping is used
- It may allow for long periods with no thrust

```
center_activate = true
hybernation_radius = 100
inner_radius  = 10
outer_radius  = 30
transit_speed = 10
outer_speed   = 30
```







END