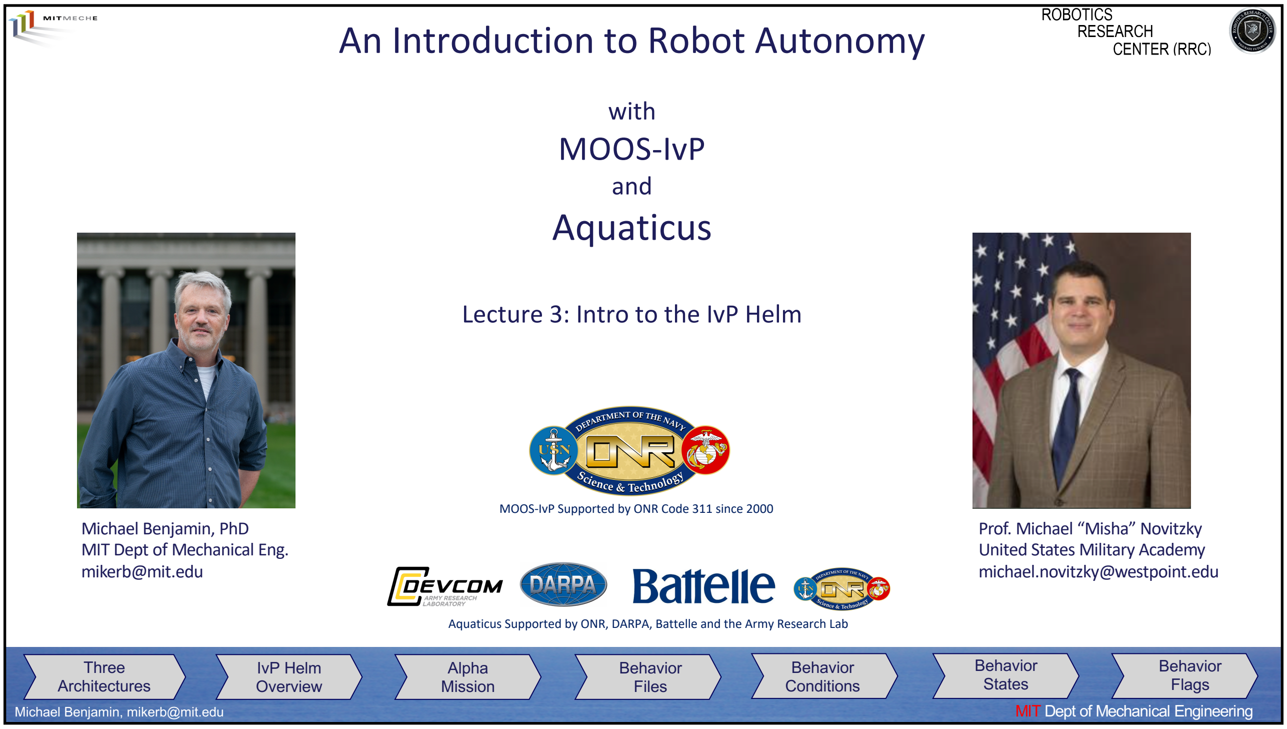


ROBOTICS
RESEARCH
CENTER (RRC)

An Introduction to Robot Autonomy

with
MOOS-IvP
and
Aquaticus

Lecture 3: Intro to the IvP Helm

Michael Benjamin, PhD
MIT Dept of Mechanical Eng.
mikerb@mit.edu

MOOS-IvP Supported by ONR Code 311 since 2000

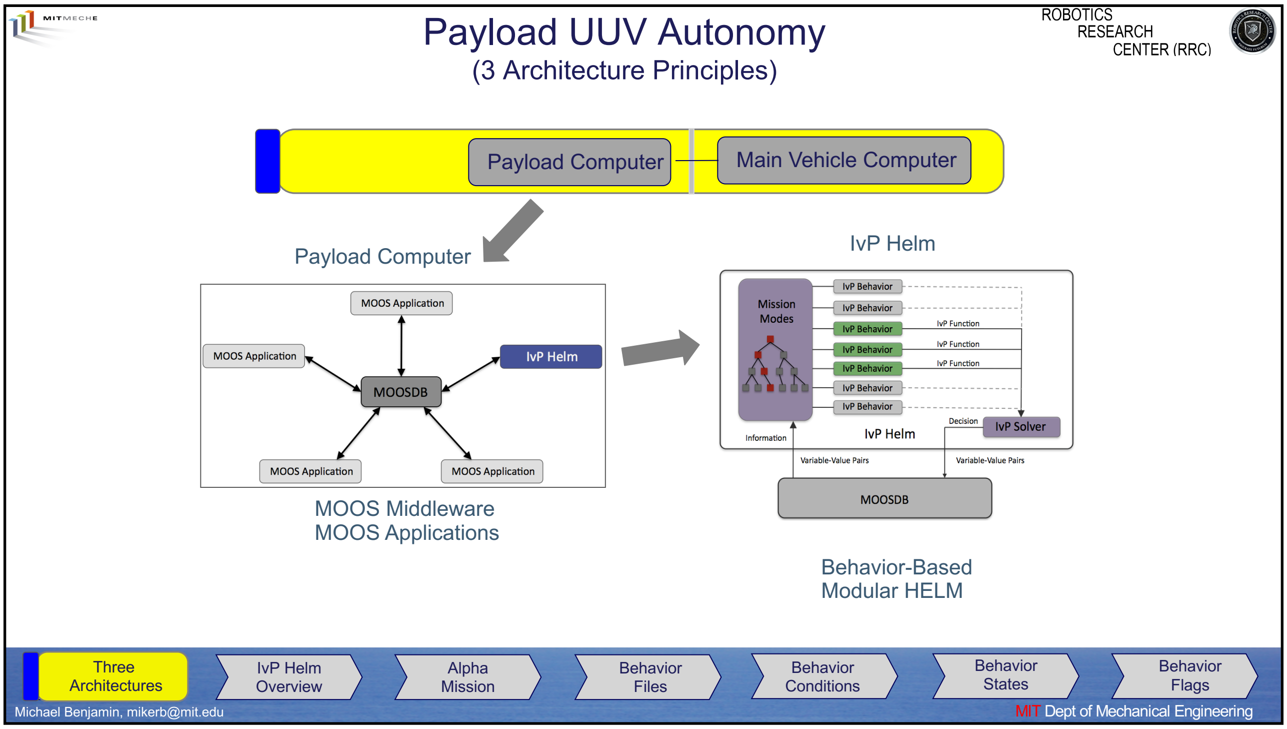
Prof. Michael "Misha" Novitzky
United States Military Academy
michael.novitzky@westpoint.edu

Aquaticus Supported by ONR, DARPA, Battelle and the Army Research Lab

Three Architectures | IvP Helm Overview | Alpha Mission | Behavior Files | Behavior Conditions | Behavior States | Behavior Flags

Michael Benjamin, mikerb@mit.edu
MIT Dept of Mechanical Engineering

1



ROBOTICS
RESEARCH
CENTER (RRC)

Payload UUV Autonomy
(3 Architecture Principles)

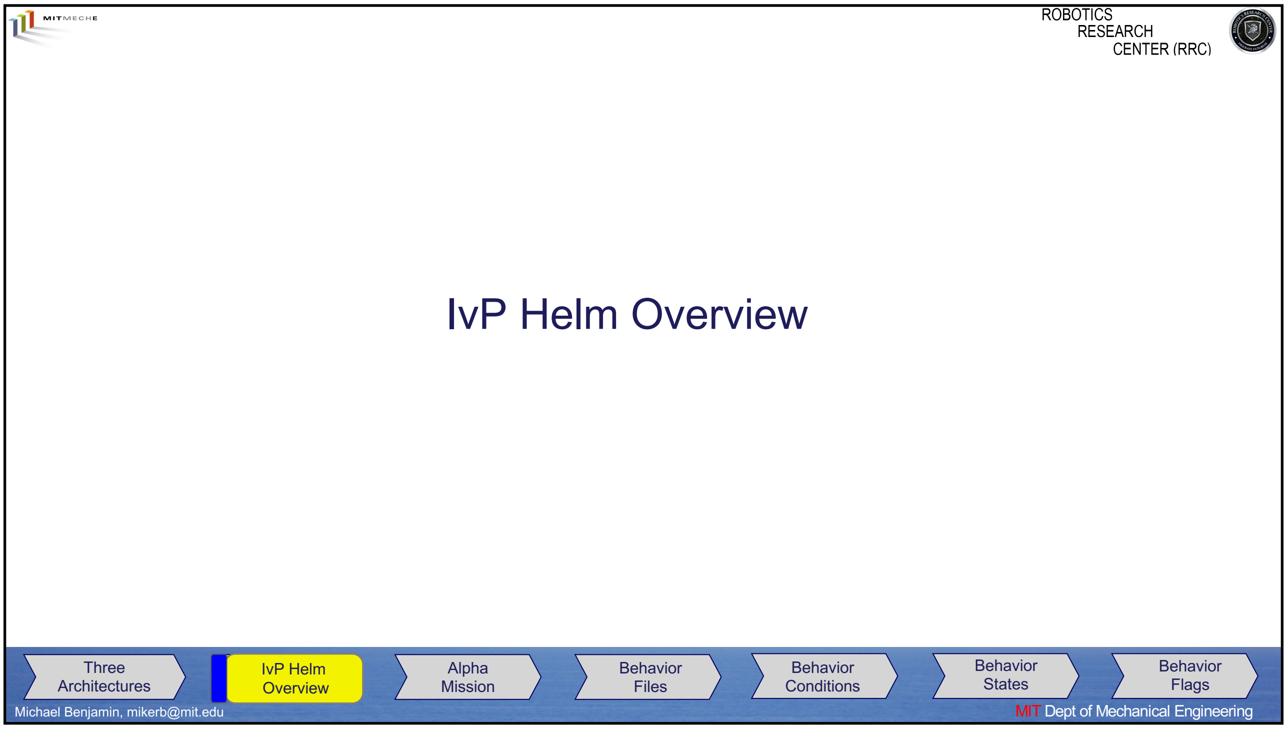
MOOS Middleware
MOOS Applications

Behavior-Based
Modular HELM

Three Architectures | IvP Helm Overview | Alpha Mission | Behavior Files | Behavior Conditions | Behavior States | Behavior Flags

Michael Benjamin, mikerb@mit.edu
MIT Dept of Mechanical Engineering

2



ROBOTICS
RESEARCH
CENTER (RRC)

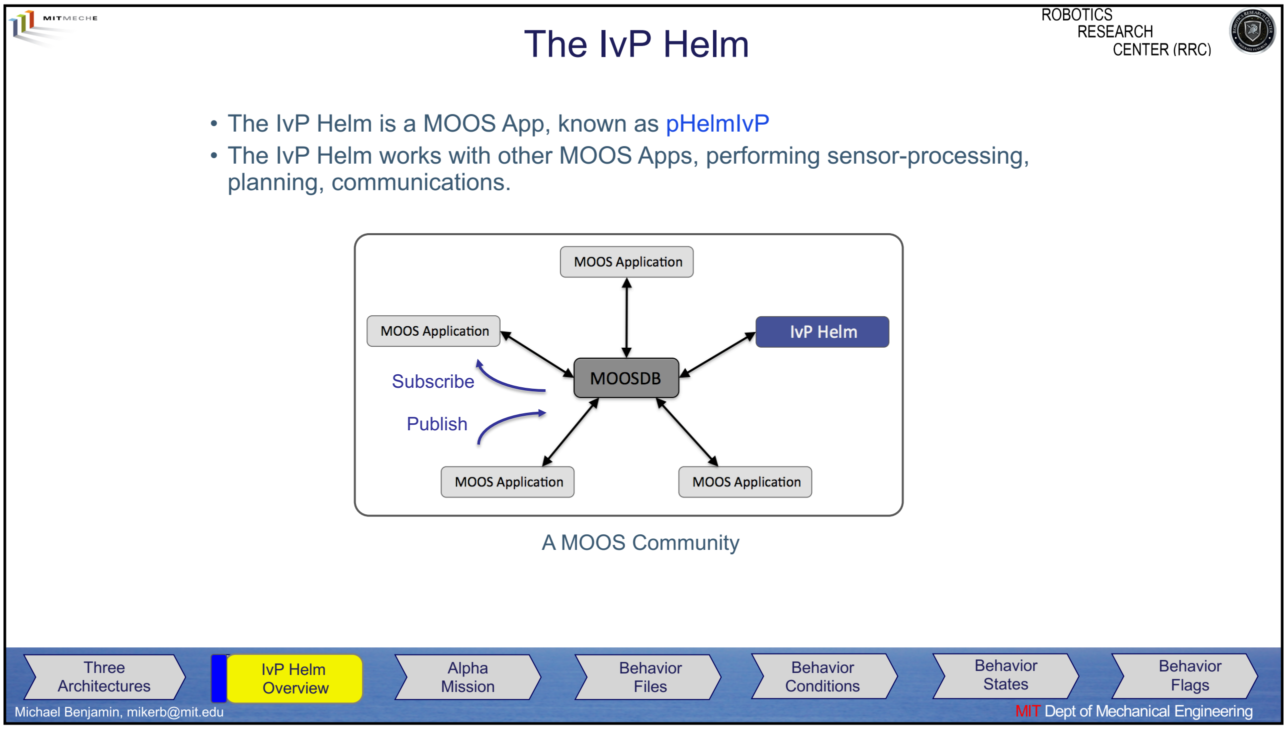
IvP Helm Overview

Three Architectures | IvP Helm Overview | Alpha Mission | Behavior Files | Behavior Conditions | Behavior States | Behavior Flags

Michael Benjamin, mikerb@mit.edu

MIT Dept of Mechanical Engineering

3



ROBOTICS
RESEARCH
CENTER (RRC)

The IvP Helm

- The IvP Helm is a MOOS App, known as pHelmIvP
- The IvP Helm works with other MOOS Apps, performing sensor-processing, planning, communications.

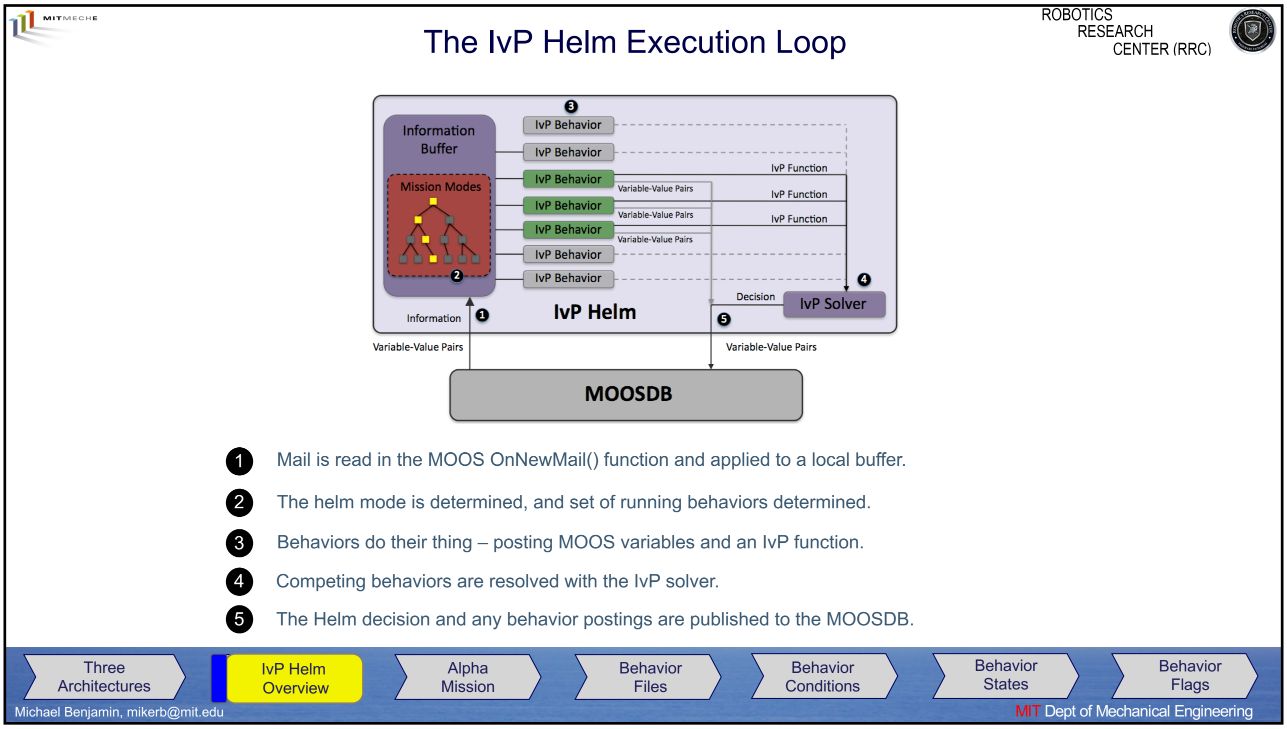
A MOOS Community

Three Architectures | IvP Helm Overview | Alpha Mission | Behavior Files | Behavior Conditions | Behavior States | Behavior Flags

Michael Benjamin, mikerb@mit.edu

MIT Dept of Mechanical Engineering

4



The IvP Helm Execution Loop

1 Mail is read in the MOOS OnNewMail() function and applied to a local buffer.

2 The helm mode is determined, and set of running behaviors determined.

3 Behaviors do their thing – posting MOOS variables and an IvP function.

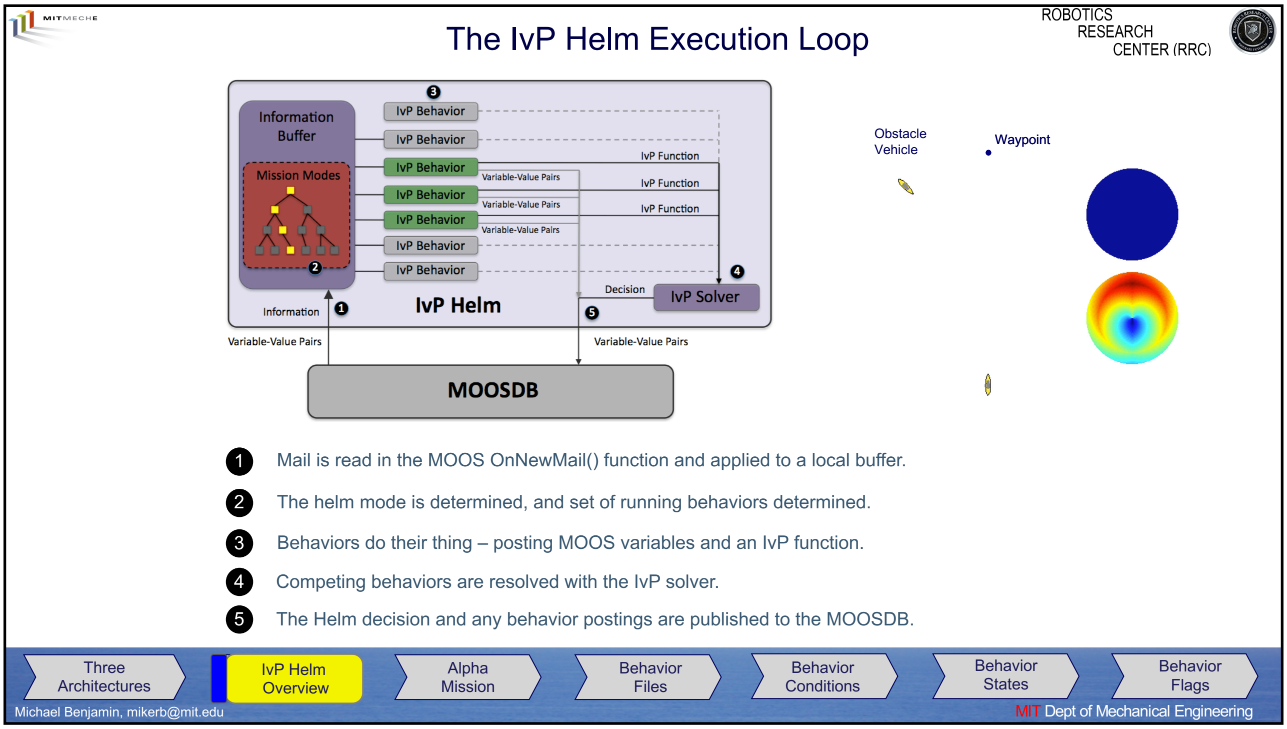
4 Competing behaviors are resolved with the IvP solver.

5 The Helm decision and any behavior postings are published to the MOOSDB.

Michael Benjamin, mikerb@mit.edu

MIT Dept of Mechanical Engineering

5



The IvP Helm Execution Loop

1 Mail is read in the MOOS OnNewMail() function and applied to a local buffer.

2 The helm mode is determined, and set of running behaviors determined.

3 Behaviors do their thing – posting MOOS variables and an IvP function.

4 Competing behaviors are resolved with the IvP solver.

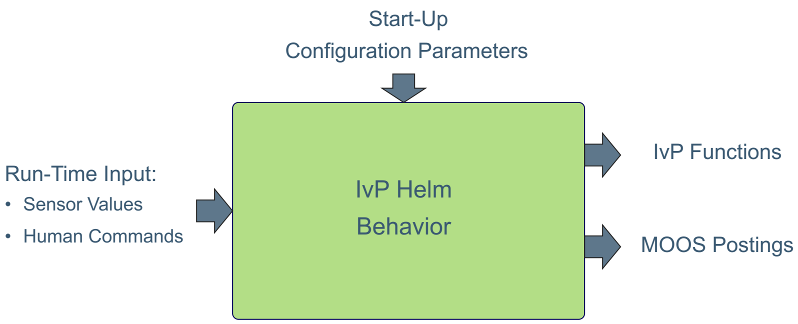
5 The Helm decision and any behavior postings are published to the MOOSDB.

Michael Benjamin, mikerb@mit.edu

MIT Dept of Mechanical Engineering

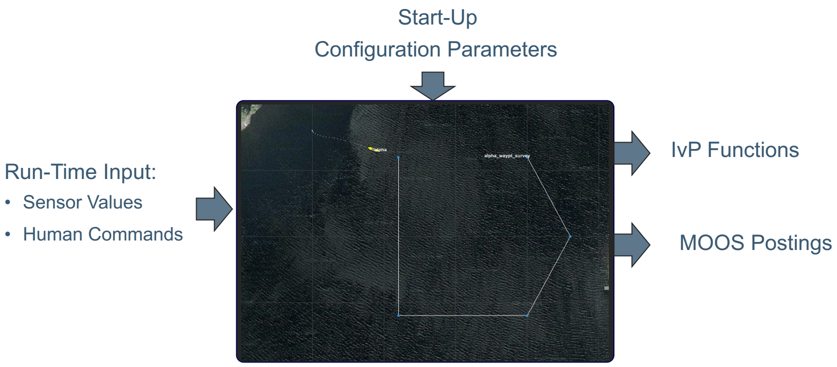
6

Behaviors Generally

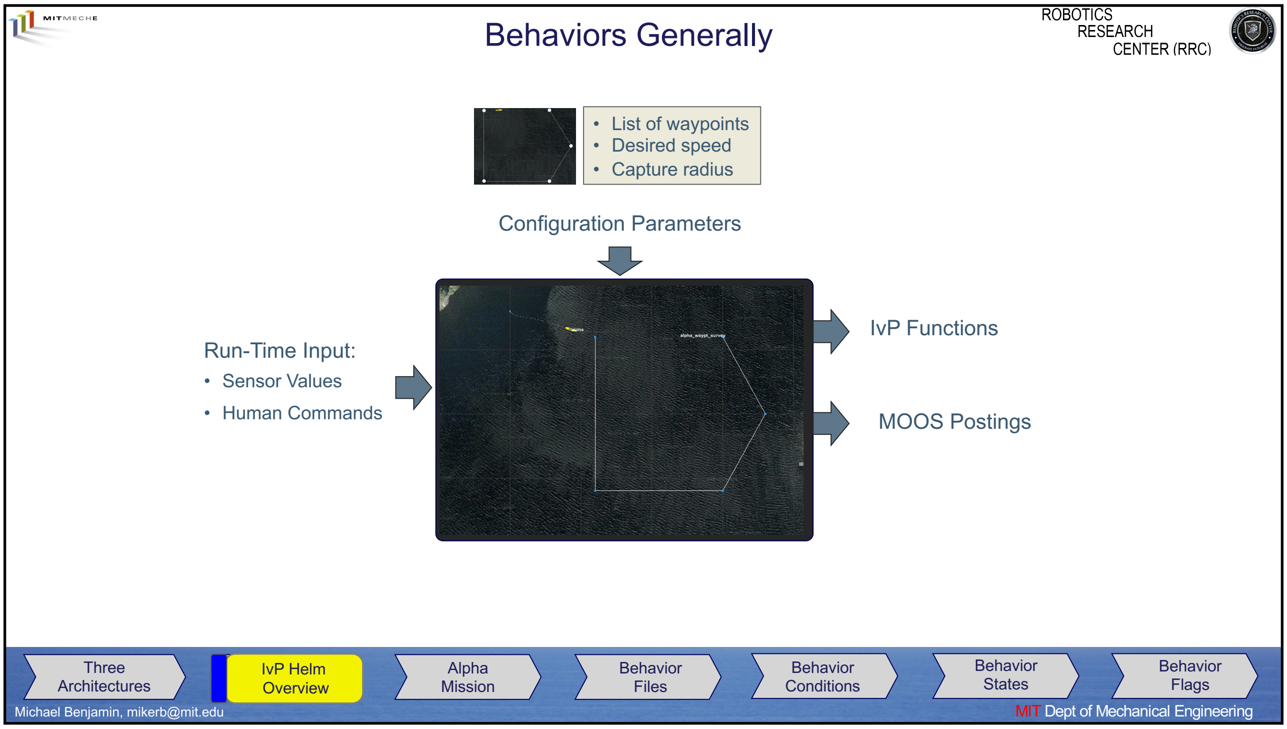


Three Architectures | IvP Helm Overview | Alpha Mission | Behavior Files | Behavior Conditions | Behavior States | Behavior Flags

Example: Waypoint Behavior



Three Architectures | IvP Helm Overview | Alpha Mission | Behavior Files | Behavior Conditions | Behavior States | Behavior Flags



ROBOTICS
RESEARCH
CENTER (RRC)

Behaviors Generally

- List of waypoints
- Desired speed
- Capture radius

Configuration Parameters

Run-Time Input:

- Sensor Values
- Human Commands

IvP Functions

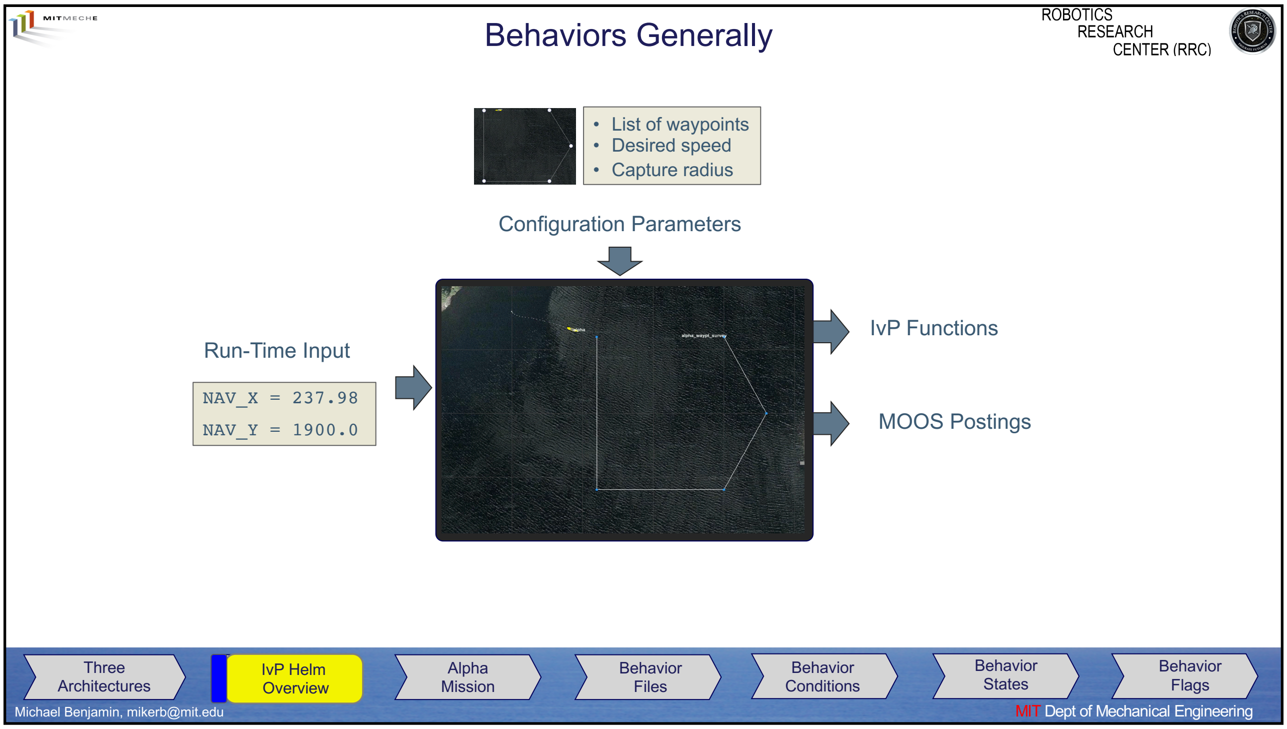
MOOS Postings

Three Architectures | IvP Helm Overview | Alpha Mission | Behavior Files | Behavior Conditions | Behavior States | Behavior Flags

Michael Benjamin, mikerb@mit.edu

MIT Dept of Mechanical Engineering

9



ROBOTICS
RESEARCH
CENTER (RRC)

Behaviors Generally

- List of waypoints
- Desired speed
- Capture radius

Configuration Parameters

Run-Time Input

```
NAV_X = 237.98
NAV_Y = 1900.0
```

IvP Functions

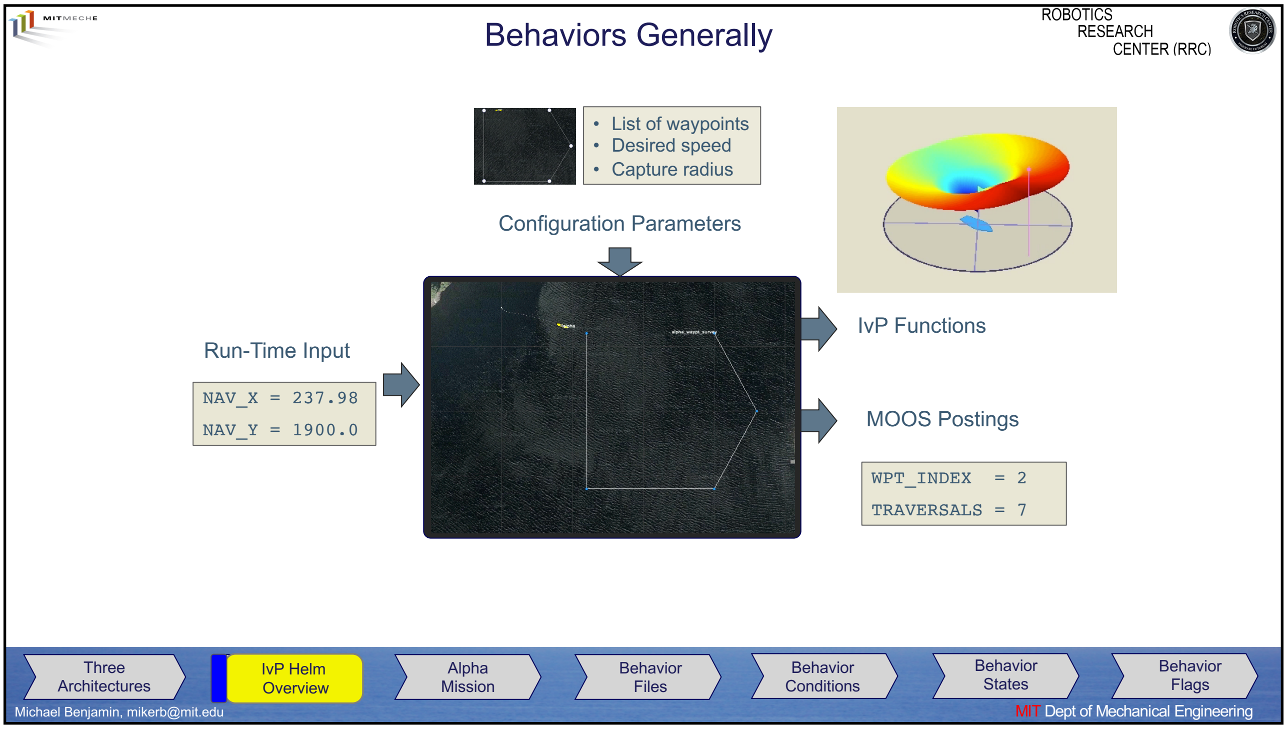
MOOS Postings

Three Architectures | IvP Helm Overview | Alpha Mission | Behavior Files | Behavior Conditions | Behavior States | Behavior Flags

Michael Benjamin, mikerb@mit.edu

MIT Dept of Mechanical Engineering

10



Behaviors Generally

- List of waypoints
- Desired speed
- Capture radius

Configuration Parameters

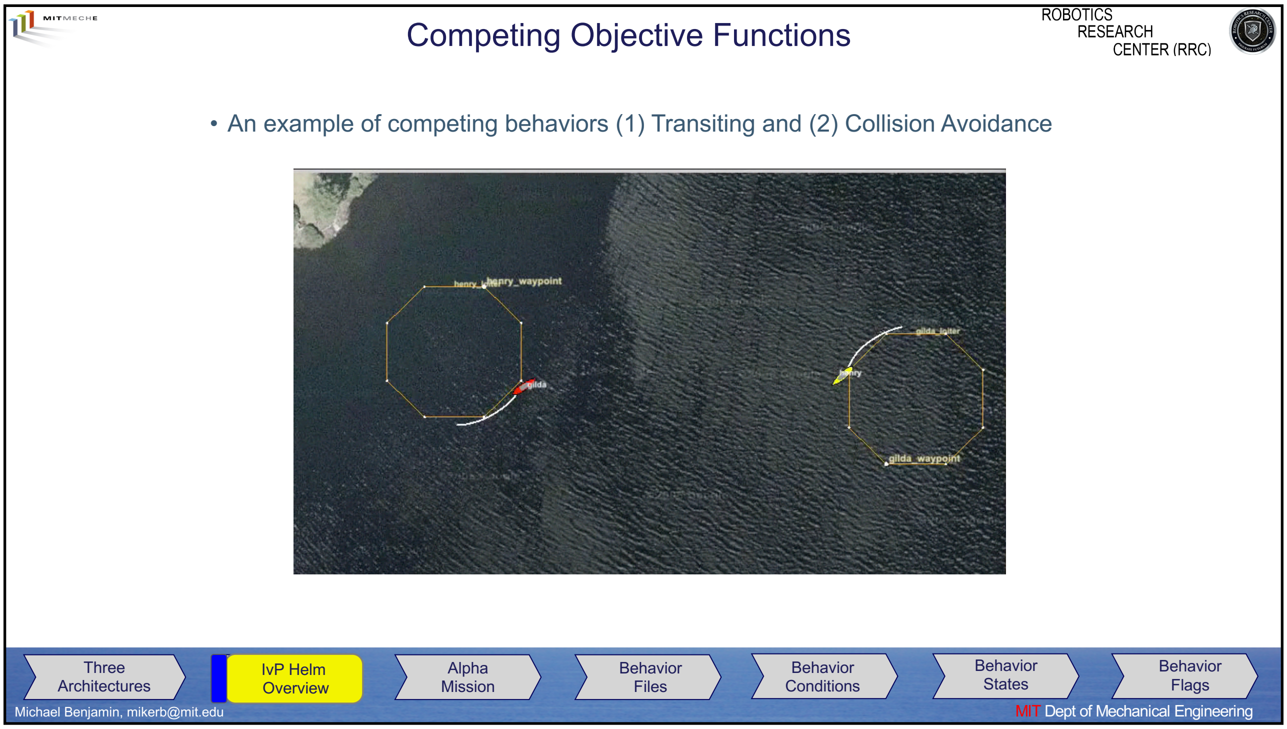
Run-Time Input

```
NAV_X = 237.98
NAV_Y = 1900.0
```

IvP Functions

MOOS Postings

```
WPT_INDEX   = 2
TRAVERSALS  = 7
```

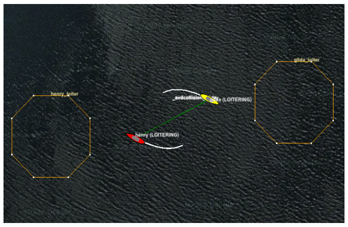


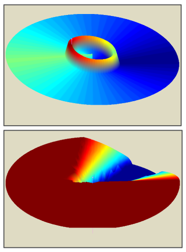
Competing Objective Functions

- An example of competing behaviors (1) Transiting and (2) Collision Avoidance

Competing Objective Functions

- Each vehicle is running two behaviors
- Each produces its own objective function



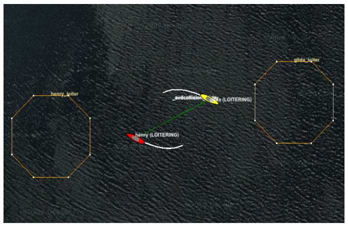


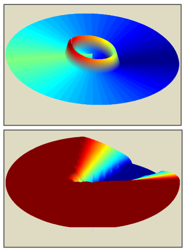
Transiting
Objective Function

Collision Avoidance
Objective Function

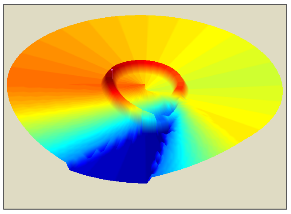
Competing Objective Functions

- Each vehicle is running two behaviors
- Each produces its own objective function

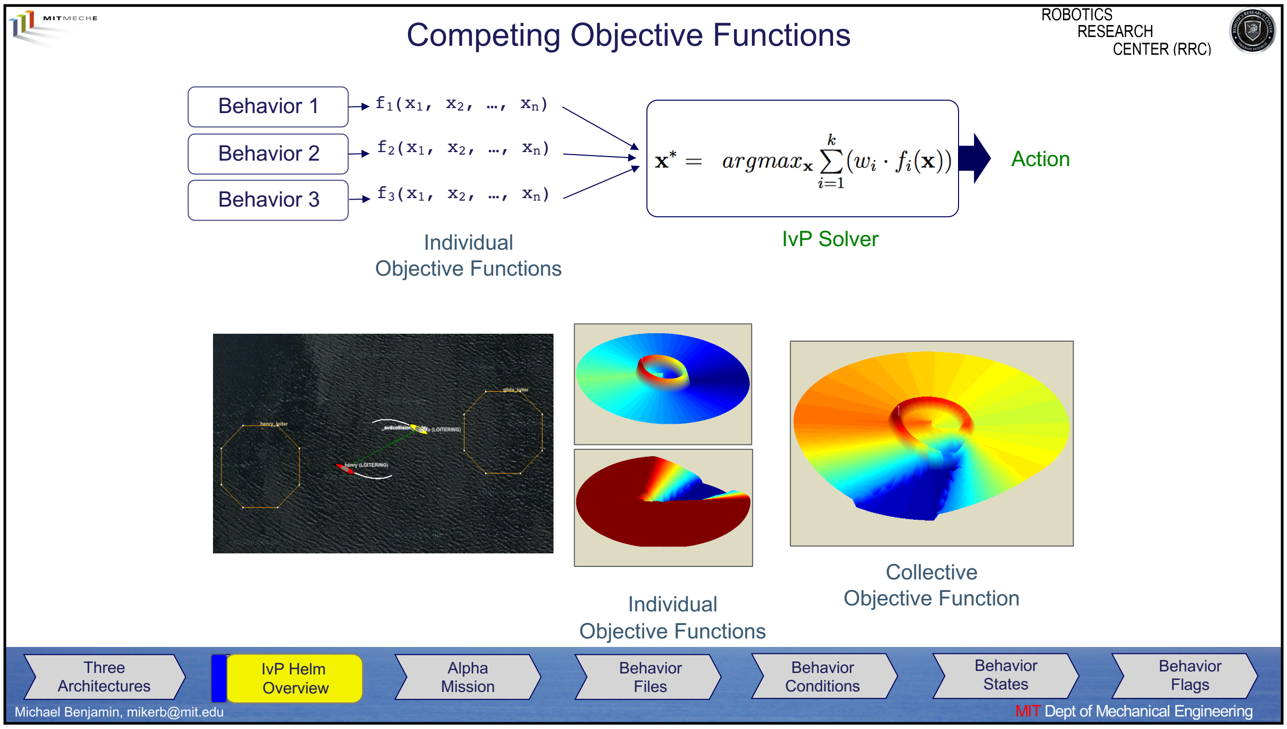




Individual
Objective Functions



Collective
Objective Function



Competing Objective Functions

ROBOTICS
RESEARCH
CENTER (RRC)

Behavior 1 → $f_1(x_1, x_2, \ldots, x_n)$

Behavior 2 → $f_2(x_1, x_2, \ldots, x_n)$

Behavior 3 → $f_3(x_1, x_2, \ldots, x_n)$

$$\mathbf{x}^* = argmax_{\mathbf{x}} \sum_{i=1}^{k} (w_i \cdot f_i(\mathbf{x}))$$

Action

Individual
Objective Functions

IvP Solver

Individual
Objective Functions

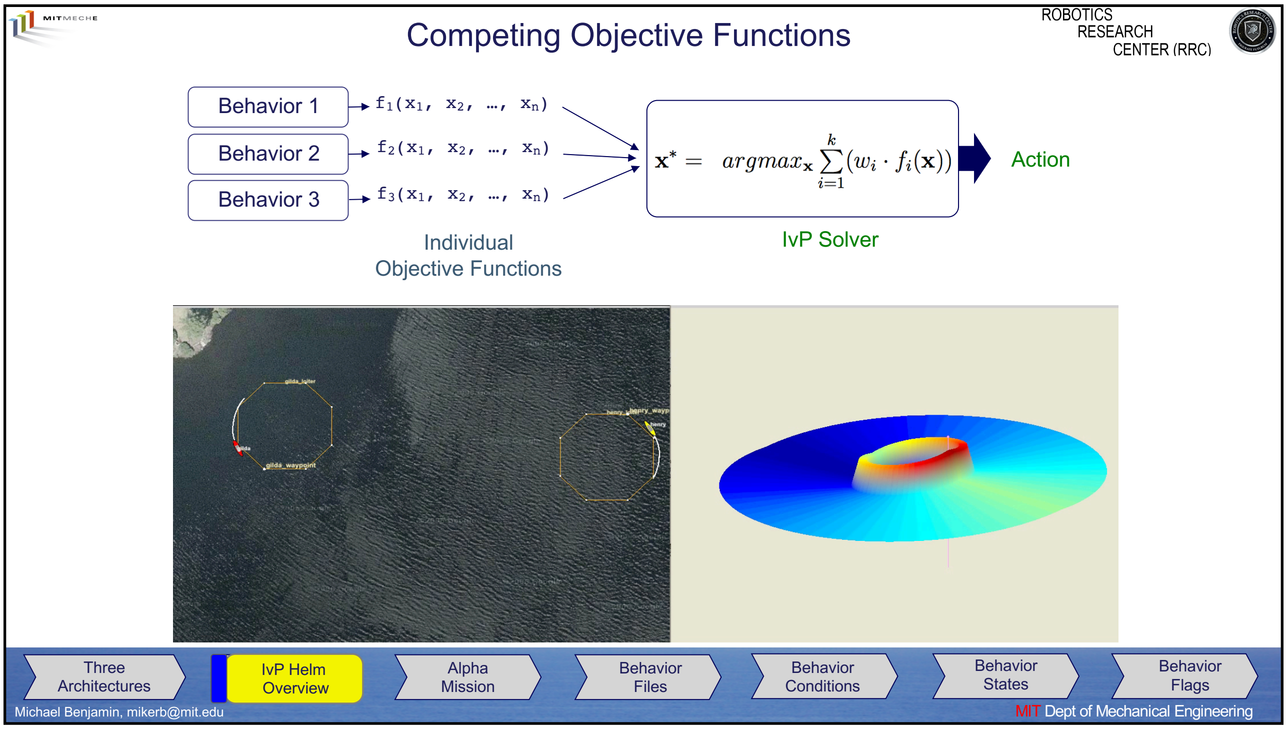
Collective
Objective Function

Three Architectures | IvP Helm Overview | Alpha Mission | Behavior Files | Behavior Conditions | Behavior States | Behavior Flags

Michael Benjamin, mikerb@mit.edu

MIT Dept of Mechanical Engineering

15



Competing Objective Functions

ROBOTICS
RESEARCH
CENTER (RRC)

Behavior 1 → $f_1(x_1, x_2, \ldots, x_n)$

Behavior 2 → $f_2(x_1, x_2, \ldots, x_n)$

Behavior 3 → $f_3(x_1, x_2, \ldots, x_n)$

$$\mathbf{x}^* = argmax_{\mathbf{x}} \sum_{i=1}^{k} (w_i \cdot f_i(\mathbf{x}))$$

Action

Individual
Objective Functions

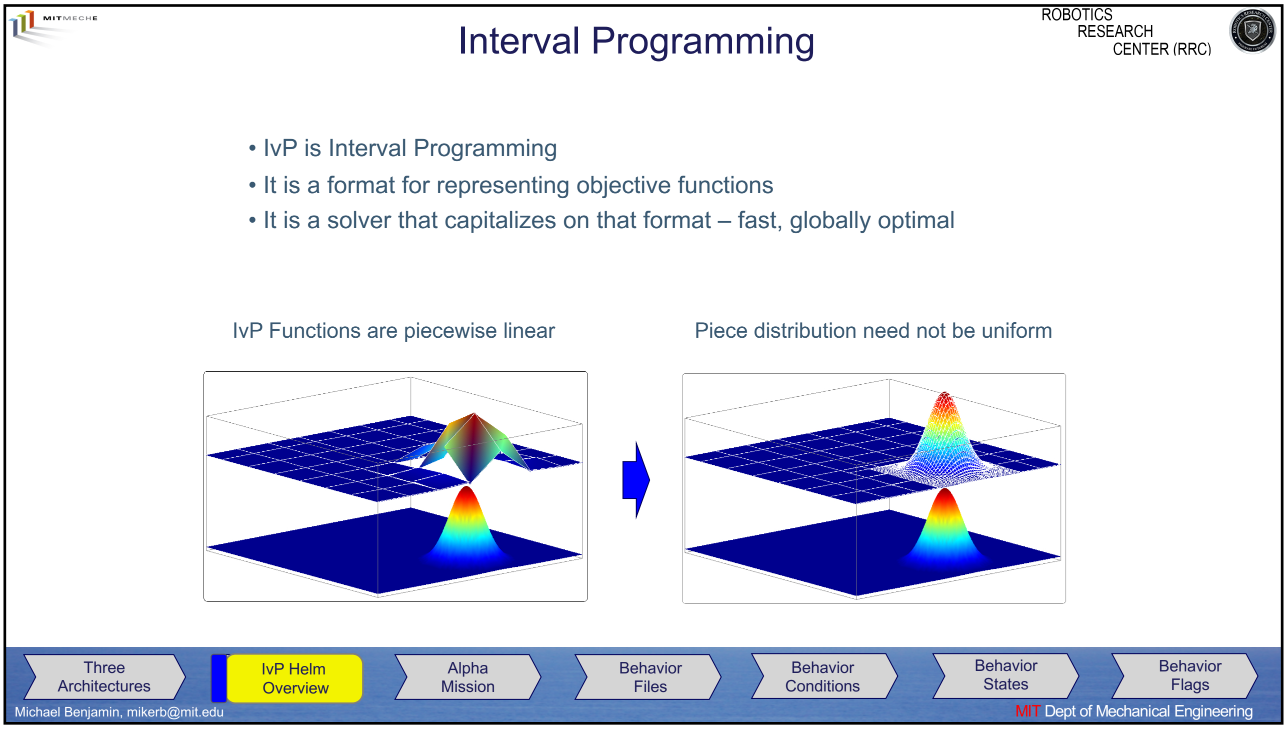
IvP Solver

Three Architectures | IvP Helm Overview | Alpha Mission | Behavior Files | Behavior Conditions | Behavior States | Behavior Flags

Michael Benjamin, mikerb@mit.edu

MIT Dept of Mechanical Engineering

16



ROBOTICS
RESEARCH
CENTER (RRC)

Interval Programming

- IvP is Interval Programming
- It is a format for representing objective functions
- It is a solver that capitalizes on that format – fast, globally optimal

IvP Functions are piecewise linear

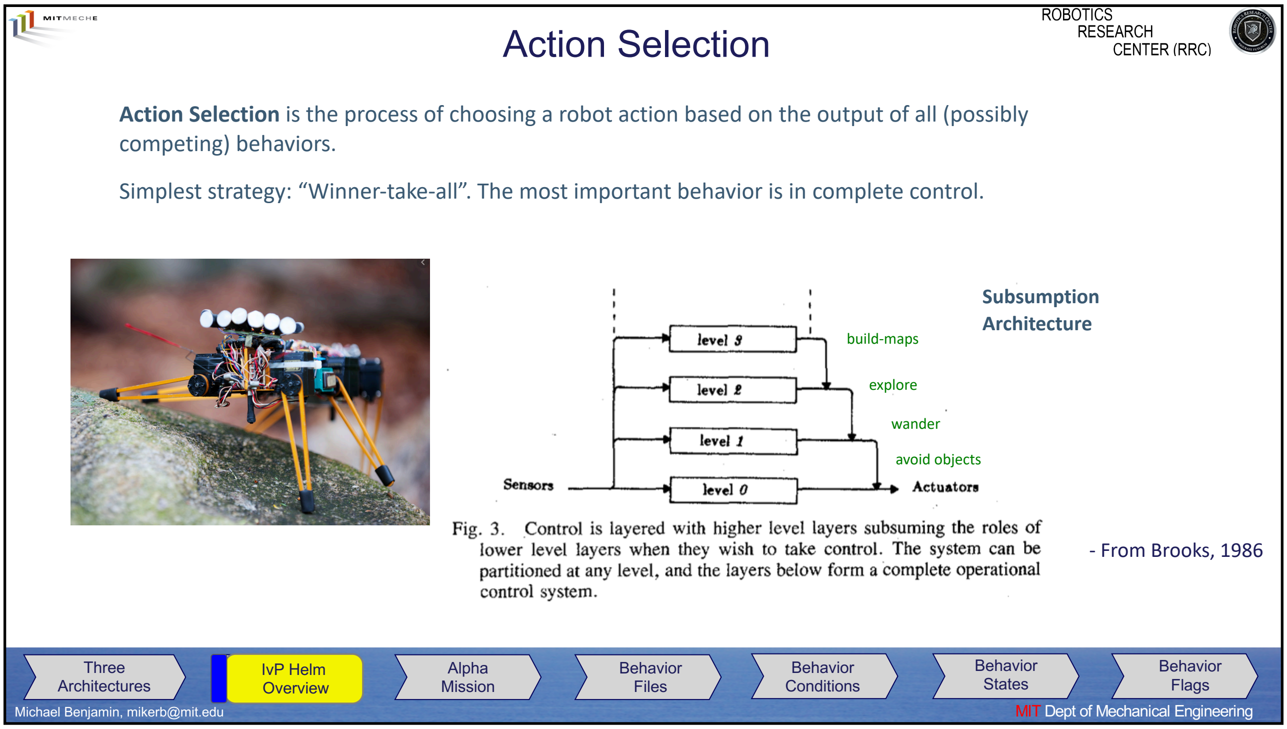
Piece distribution need not be uniform

Three Architectures | IvP Helm Overview | Alpha Mission | Behavior Files | Behavior Conditions | Behavior States | Behavior Flags

Michael Benjamin, mikerb@mit.edu

MIT Dept of Mechanical Engineering

17



ROBOTICS
RESEARCH
CENTER (RRC)

Action Selection

Action Selection is the process of choosing a robot action based on the output of all (possibly competing) behaviors.

Simplest strategy: “Winner-take-all”. The most important behavior is in complete control.

level 3
level 2
level 1
level 0
Sensors
Actuators
build-maps
explore
wander
avoid objects

Subsumption Architecture

Fig. 3. Control is layered with higher level layers subsuming the roles of lower level layers when they wish to take control. The system can be partitioned at any level, and the layers below form a complete operational control system.

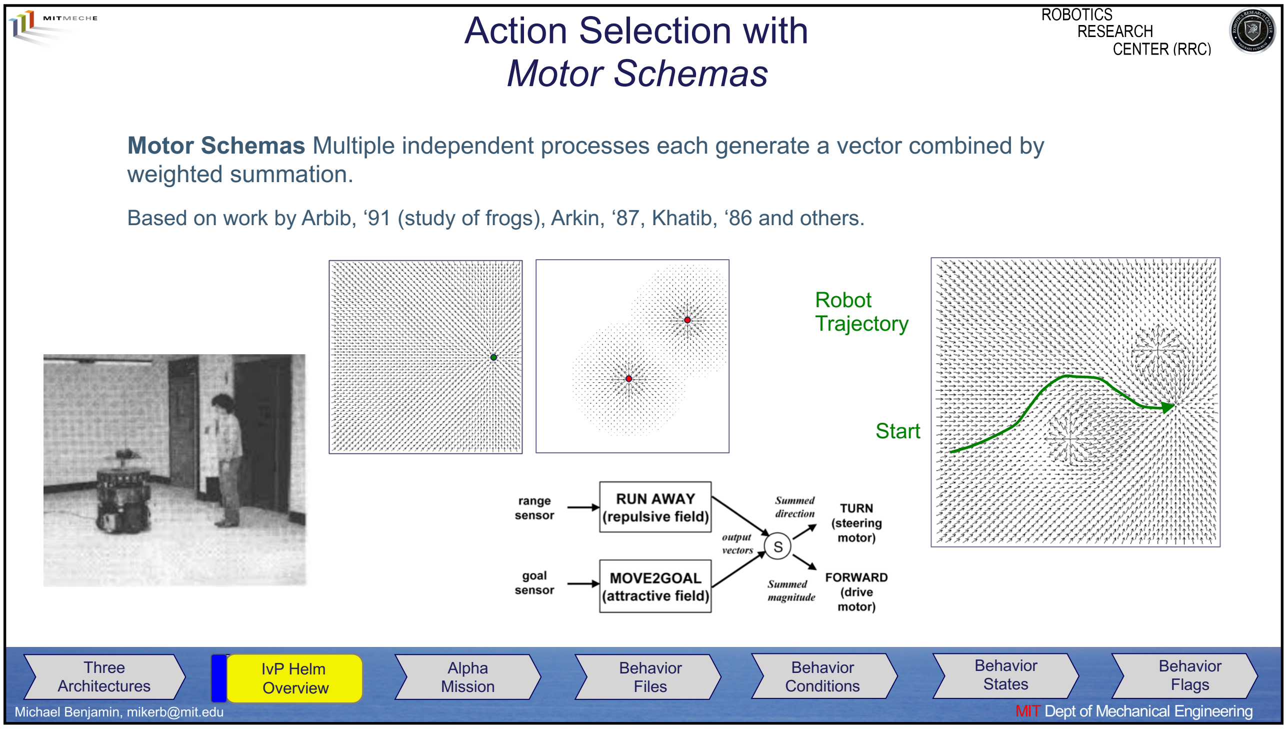
- From Brooks, 1986

Three Architectures | IvP Helm Overview | Alpha Mission | Behavior Files | Behavior Conditions | Behavior States | Behavior Flags

Michael Benjamin, mikerb@mit.edu

MIT Dept of Mechanical Engineering

18

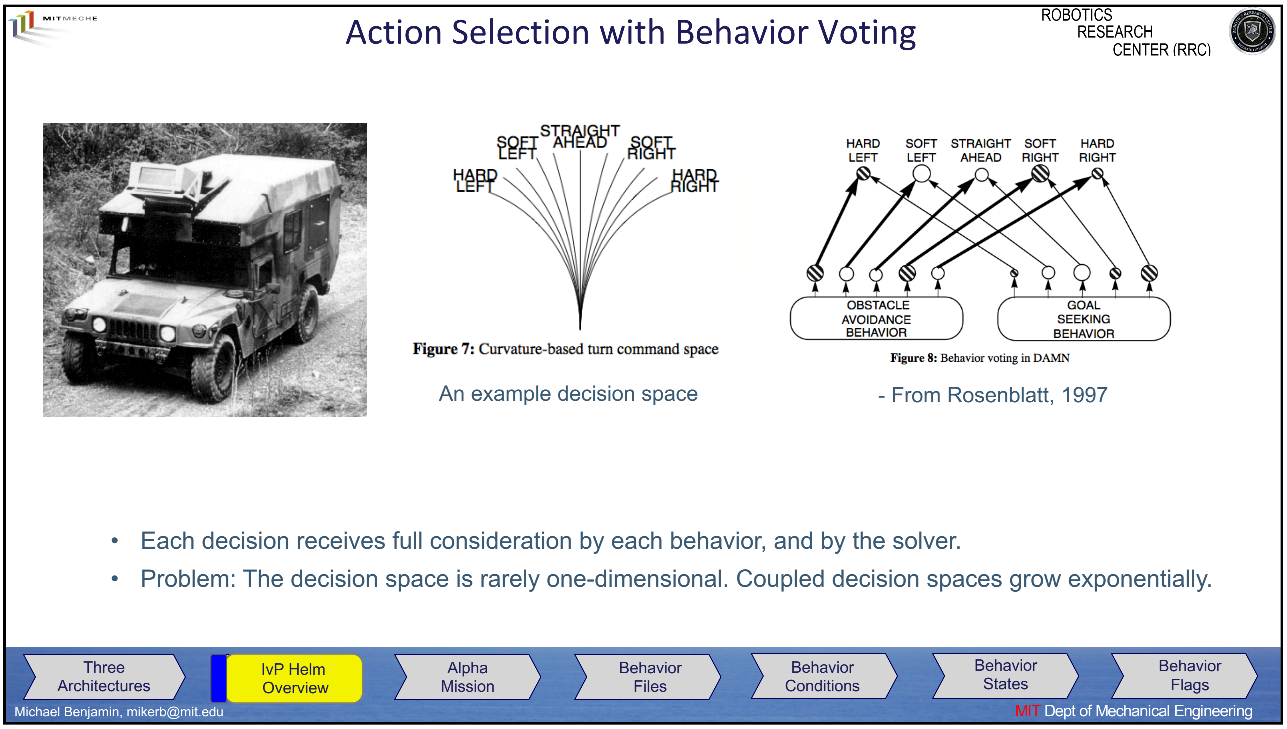


ROBOTICS
RESEARCH
CENTER (RRC)

Action Selection with *Motor Schemas*

Motor Schemas Multiple independent processes each generate a vector combined by weighted summation.

Based on work by Arbib, '91 (study of frogs), Arkin, '87, Khatib, '86 and others.



ROBOTICS
RESEARCH
CENTER (RRC)

Action Selection with Behavior Voting

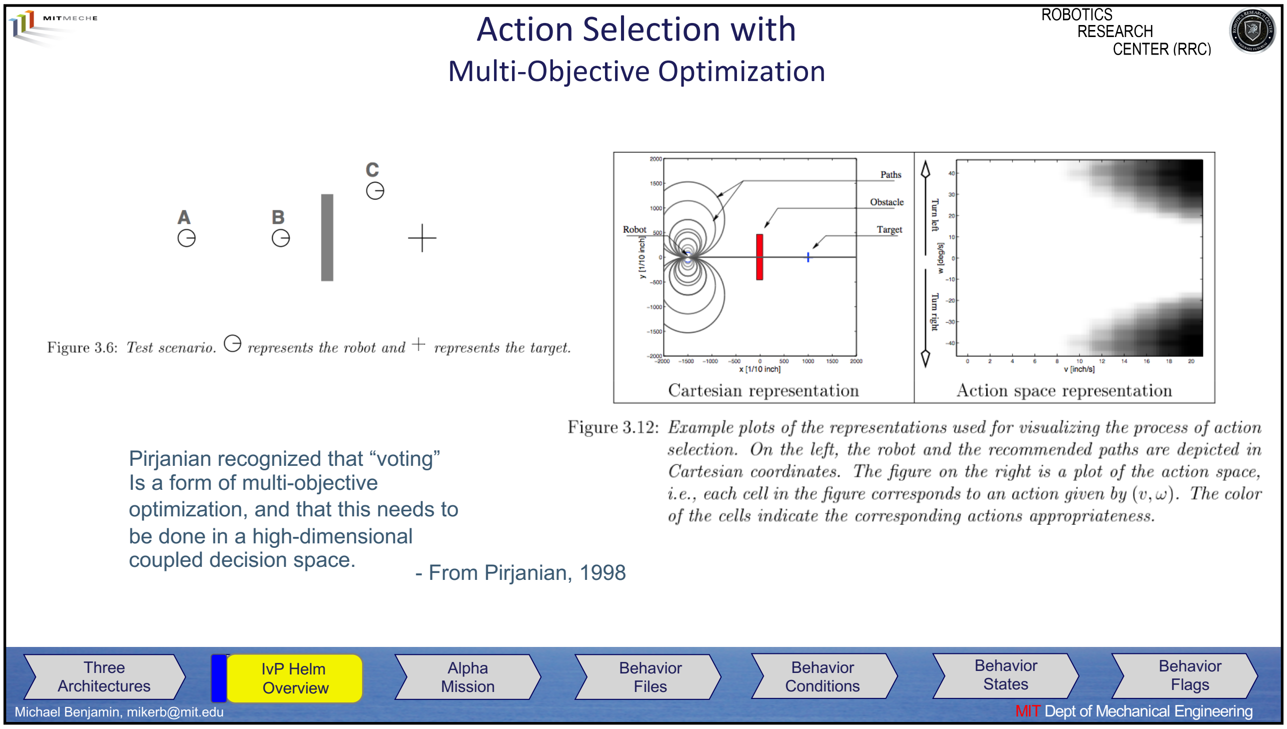
Figure 7: Curvature-based turn command space

An example decision space

Figure 8: Behavior voting in DAMN

- From Rosenblatt, 1997

- Each decision receives full consideration by each behavior, and by the solver.
- Problem: The decision space is rarely one-dimensional. Coupled decision spaces grow exponentially.



Action Selection with
Multi-Objective Optimization

ROBOTICS
RESEARCH
CENTER (RRC)

A
B
C

Figure 3.6: *Test scenario.* ⊖ *represents the robot and* + *represents the target.*

Robot
Paths
Obstacle
Target
y [1/10 inch]
x [1/10 inch]
Turn left
Turn right
w [deg/s]
v [inch/s]
Cartesian representation
Action space representation

Figure 3.12: *Example plots of the representations used for visualizing the process of action selection. On the left, the robot and the recommended paths are depicted in Cartesian coordinates. The figure on the right is a plot of the action space, i.e., each cell in the figure corresponds to an action given by (v, ω). The color of the cells indicate the corresponding actions appropriateness.*

Pirjanian recognized that "voting"
Is a form of multi-objective
optimization, and that this needs to
be done in a high-dimensional
coupled decision space.

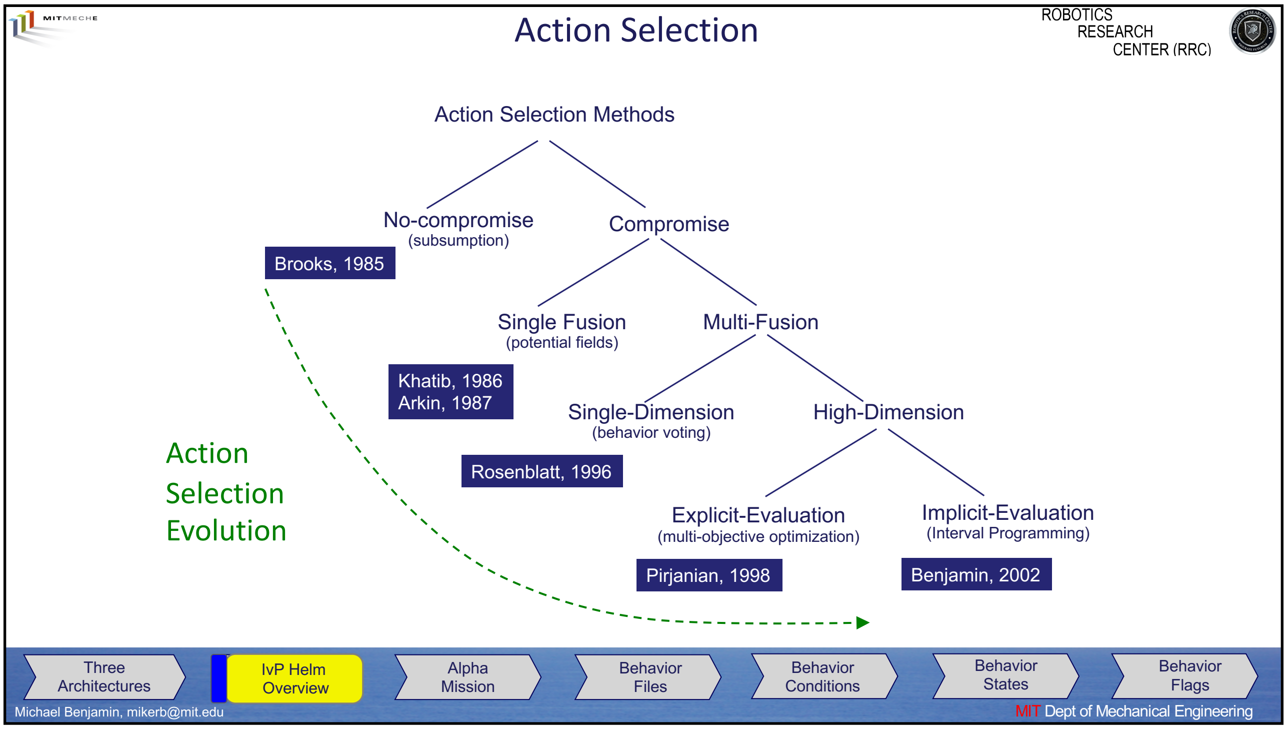
- From Pirjanian, 1998

Three Architectures | IvP Helm Overview | Alpha Mission | Behavior Files | Behavior Conditions | Behavior States | Behavior Flags

Michael Benjamin, mikerb@mit.edu

MIT Dept of Mechanical Engineering

21



Action Selection

ROBOTICS
RESEARCH
CENTER (RRC)

Action Selection Methods

No-compromise
(subsumption)

Brooks, 1985

Compromise

Single Fusion
(potential fields)

Khatib, 1986
Arkin, 1987

Multi-Fusion

Single-Dimension
(behavior voting)

Rosenblatt, 1996

High-Dimension

Explicit-Evaluation
(multi-objective optimization)

Pirjanian, 1998

Implicit-Evaluation
(Interval Programming)

Benjamin, 2002

Action
Selection
Evolution

Three Architectures | IvP Helm Overview | Alpha Mission | Behavior Files | Behavior Conditions | Behavior States | Behavior Flags

Michael Benjamin, mikerb@mit.edu

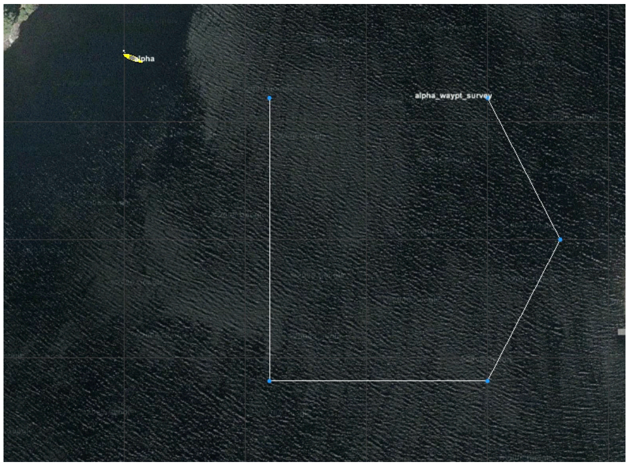
MIT Dept of Mechanical Engineering

22

The Alpha Mission

23

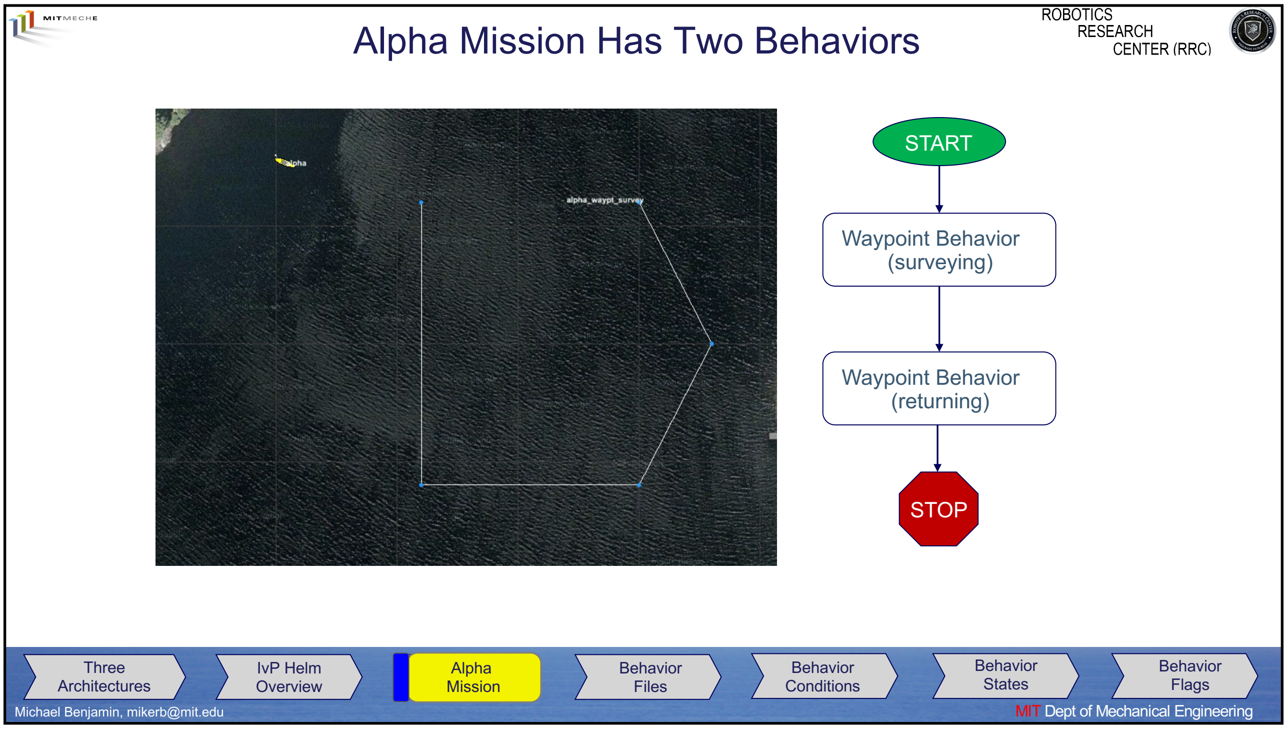
The Alpha Mission



To launch yourself:

```
$ cd moos-ivp/ivp/missions/s1_alpha
$ ./launch.sh 10
```

24



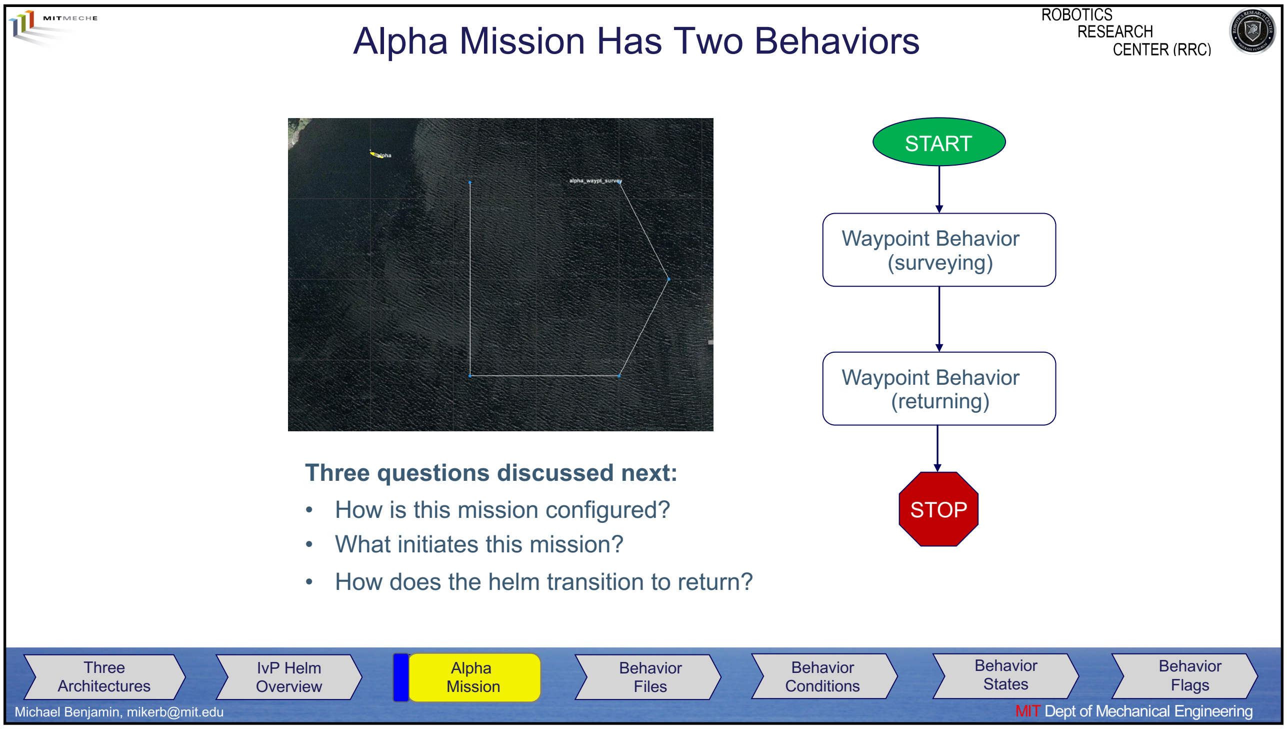
Alpha Mission Has Two Behaviors

ROBOTICS
RESEARCH
CENTER (RRC)

Michael Benjamin, mikerb@mit.edu

MIT Dept of Mechanical Engineering

25



Alpha Mission Has Two Behaviors

ROBOTICS
RESEARCH
CENTER (RRC)

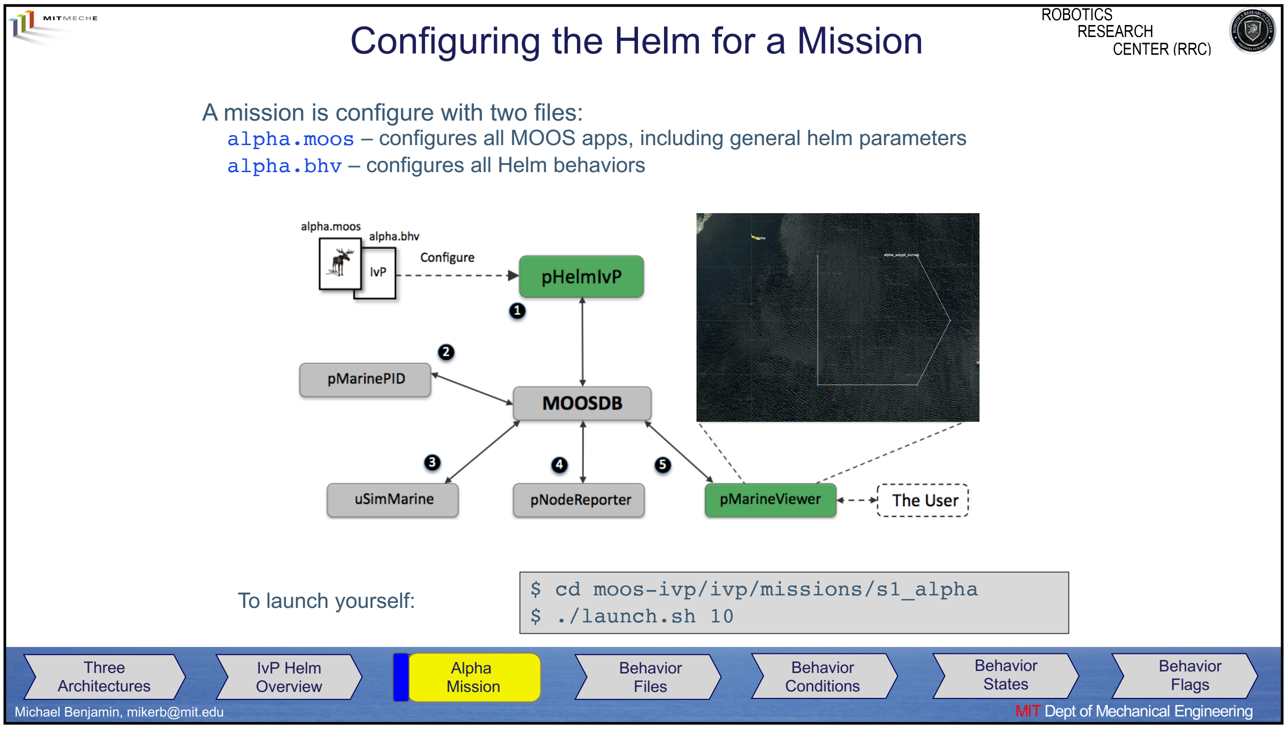
Three questions discussed next:

- How is this mission configured?
- What initiates this mission?
- How does the helm transition to return?

Michael Benjamin, mikerb@mit.edu

MIT Dept of Mechanical Engineering

26



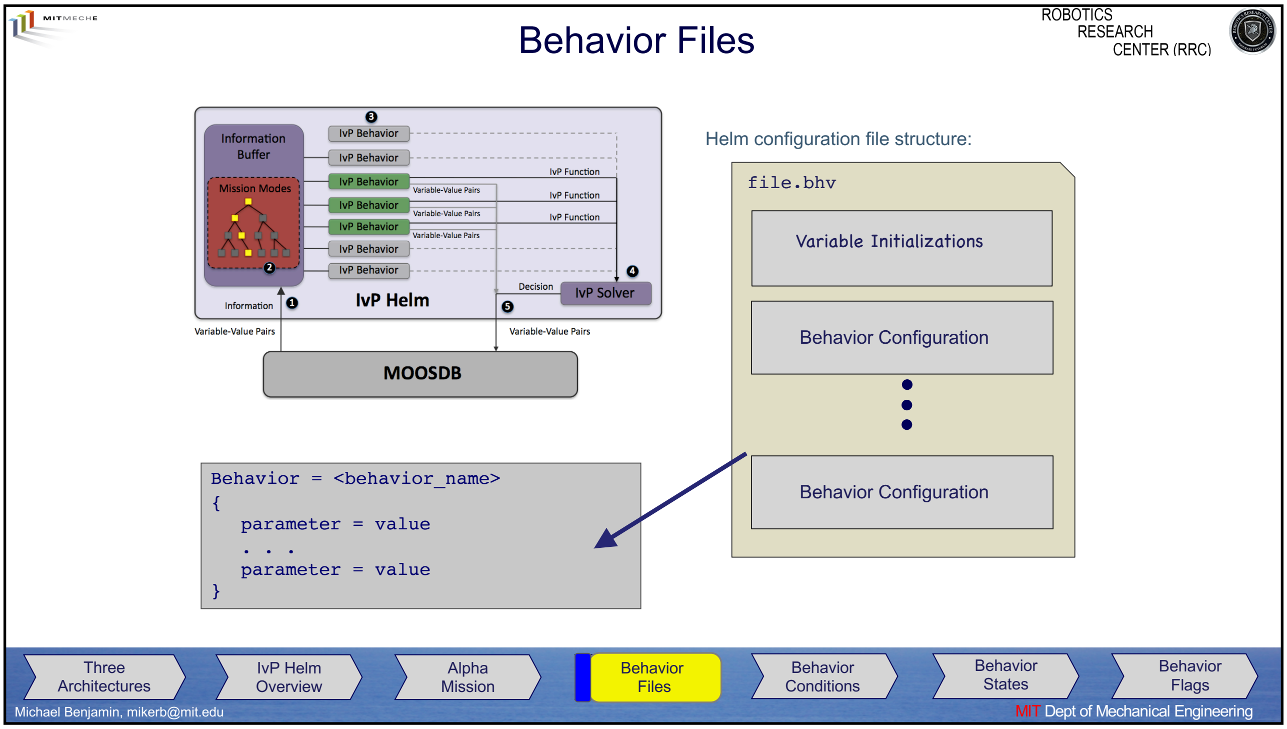
Configuring the Helm for a Mission

A mission is configure with two files:

- `alpha.moos` – configures all MOOS apps, including general helm parameters
- `alpha.bhv` – configures all Helm behaviors

To launch yourself:

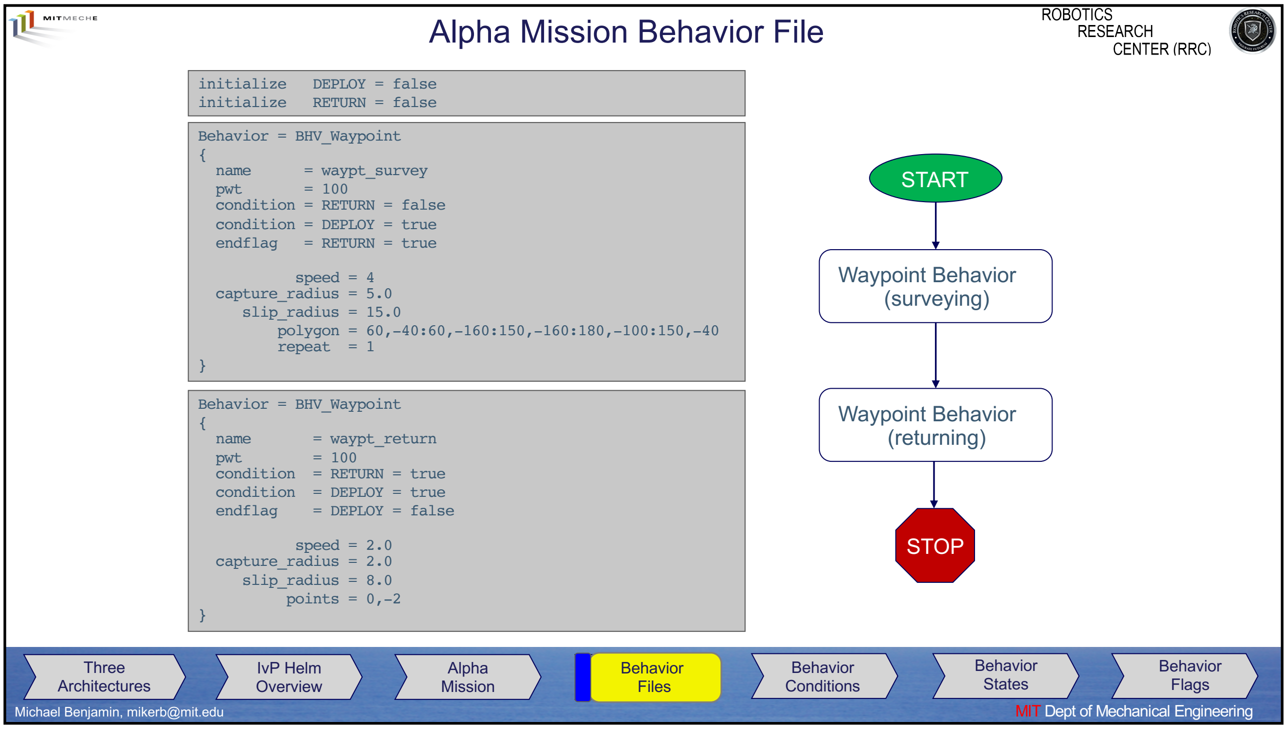
```
$ cd moos-ivp/ivp/missions/s1_alpha
$ ./launch.sh 10
```



Behavior Files

Helm configuration file structure:

```
Behavior = <behavior_name>
{
   parameter = value
   . . .
   parameter = value
}
```



MITMECHE
ROBOTICS
RESEARCH
CENTER (RRC)

Alpha Mission Behavior File

```
initialize   DEPLOY = false
initialize   RETURN = false
```

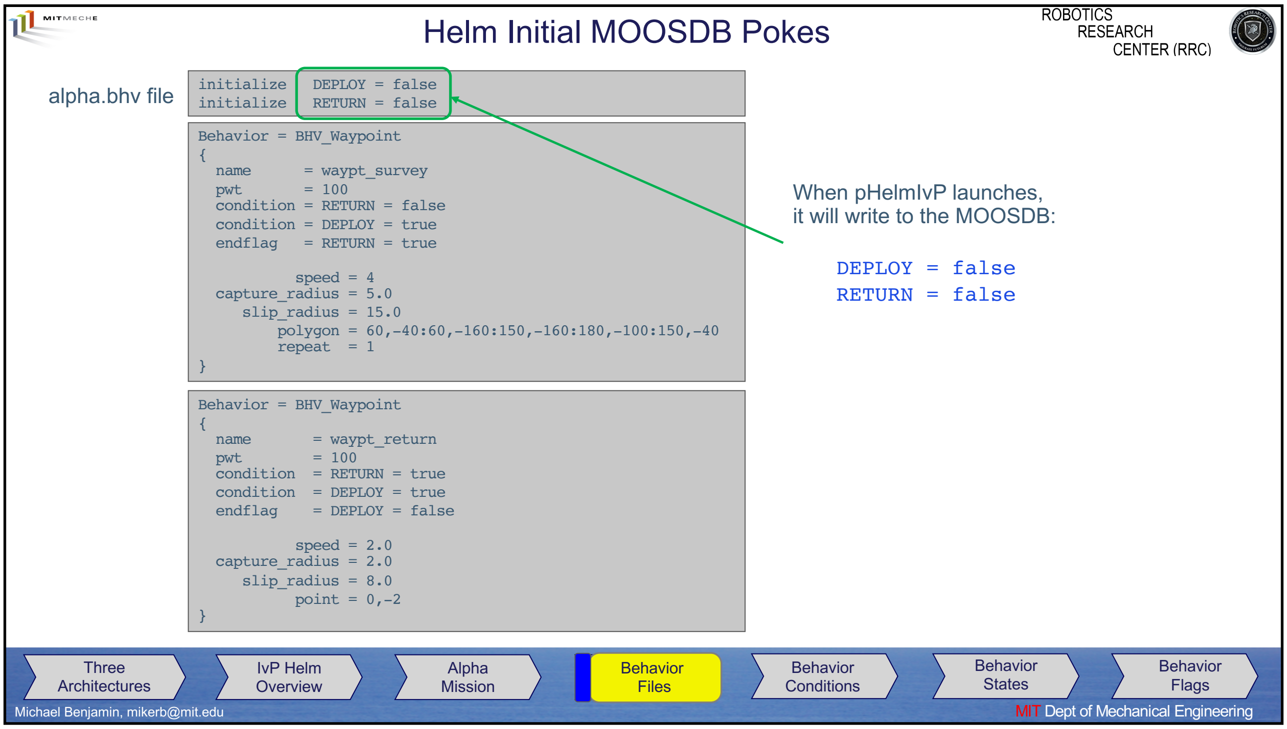
```
Behavior = BHV_Waypoint
{
  name      = waypt_survey
  pwt       = 100
  condition = RETURN = false
  condition = DEPLOY = true
  endflag   = RETURN = true

           speed = 4
  capture_radius = 5.0
     slip_radius = 15.0
         polygon = 60,-40:60,-160:150,-160:180,-100:150,-40
         repeat  = 1
}
```

```
Behavior = BHV_Waypoint
{
  name       = waypt_return
  pwt        = 100
  condition  = RETURN = true
  condition  = DEPLOY = true
  endflag    = DEPLOY = false

           speed = 2.0
  capture_radius = 2.0
     slip_radius = 8.0
          points = 0,-2
}
```

Michael Benjamin, mikerb@mit.edu

MIT Dept of Mechanical Engineering

29



MITMECHE
ROBOTICS
RESEARCH
CENTER (RRC)

Helm Initial MOOSDB Pokes

alpha.bhv file

```
initialize   DEPLOY = false
initialize   RETURN = false
```

```
Behavior = BHV_Waypoint
{
  name      = waypt_survey
  pwt       = 100
  condition = RETURN = false
  condition = DEPLOY = true
  endflag   = RETURN = true

           speed = 4
  capture_radius = 5.0
     slip_radius = 15.0
         polygon = 60,-40:60,-160:150,-160:180,-100:150,-40
         repeat  = 1
}
```

```
Behavior = BHV_Waypoint
{
  name       = waypt_return
  pwt        = 100
  condition  = RETURN = true
  condition  = DEPLOY = true
  endflag    = DEPLOY = false

           speed = 2.0
  capture_radius = 2.0
     slip_radius = 8.0
           point = 0,-2
}
```

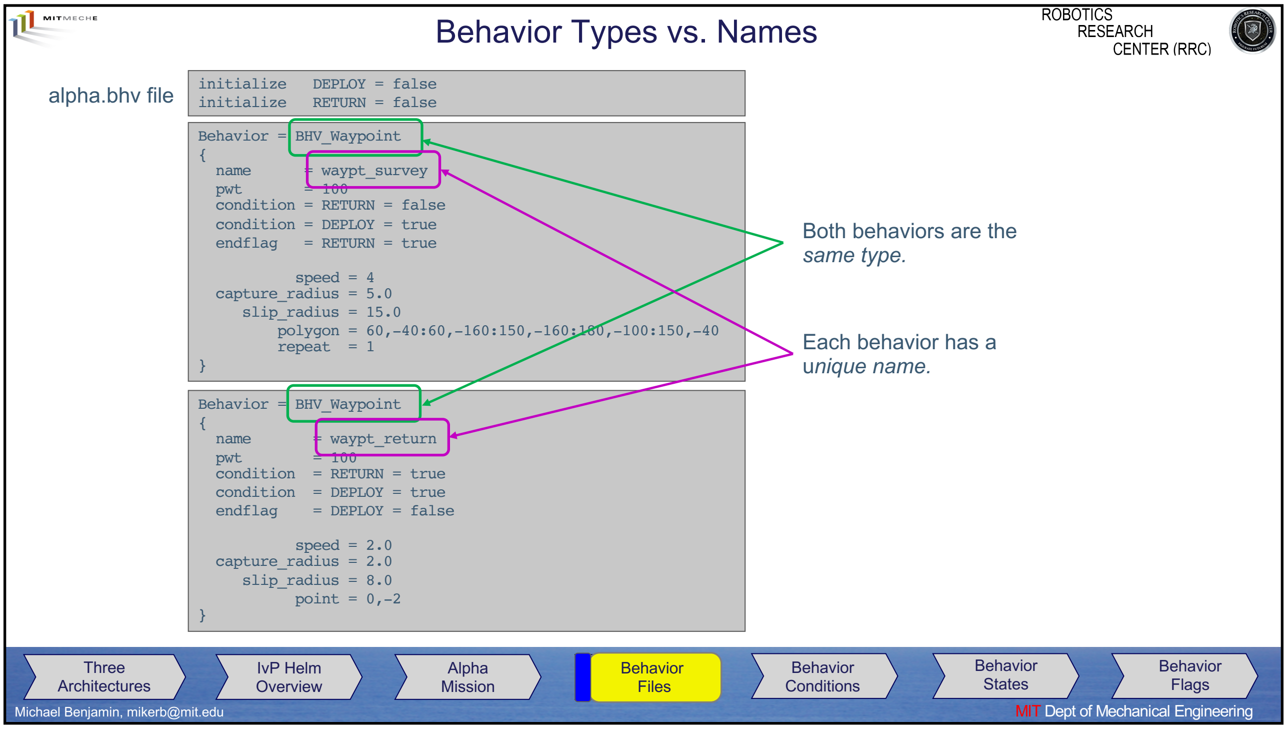
When pHelmIvP launches,
it will write to the MOOSDB:

```
DEPLOY = false
RETURN = false
```

Michael Benjamin, mikerb@mit.edu

MIT Dept of Mechanical Engineering

30



MITMECHE

ROBOTICS
RESEARCH
CENTER (RRC)

Behavior Types vs. Names

alpha.bhv file

```
initialize   DEPLOY = false
initialize   RETURN = false
```

```
Behavior = BHV_Waypoint
{
  name      = waypt_survey
  pwt       = 100
  condition = RETURN = false
  condition = DEPLOY = true
  endflag   = RETURN = true

           speed = 4
  capture_radius = 5.0
     slip_radius = 15.0
         polygon = 60,-40:60,-160:150,-160:180,-100:150,-40
         repeat  = 1
}
```

```
Behavior = BHV_Waypoint
{
  name       = waypt_return
  pwt        = 100
  condition  = RETURN = true
  condition  = DEPLOY = true
  endflag    = DEPLOY = false

           speed = 2.0
  capture_radius = 2.0
     slip_radius = 8.0
           point = 0,-2
}
```

Both behaviors are the *same type.*

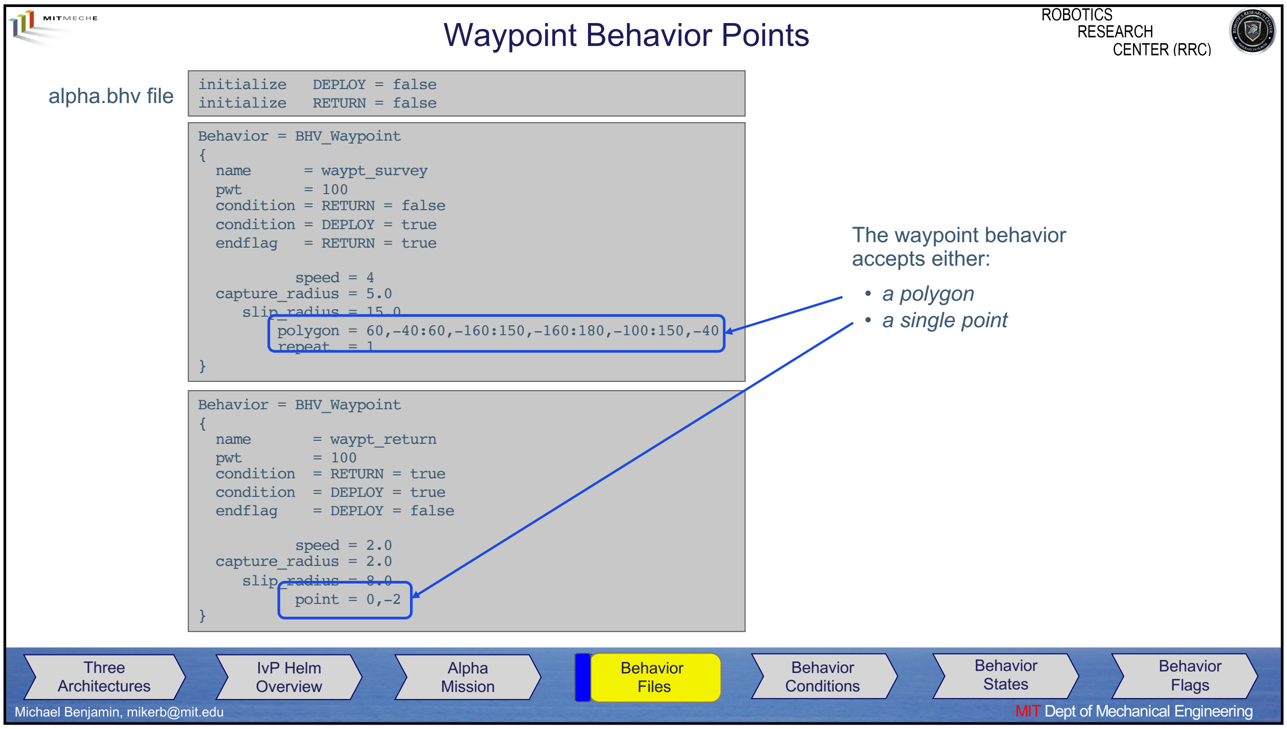
Each behavior has a *unique name.*

Three Architectures | IvP Helm Overview | Alpha Mission | Behavior Files | Behavior Conditions | Behavior States | Behavior Flags

Michael Benjamin, mikerb@mit.edu

MIT Dept of Mechanical Engineering

31



MITMECHE

ROBOTICS
RESEARCH
CENTER (RRC)

Waypoint Behavior Points

alpha.bhv file

```
initialize   DEPLOY = false
initialize   RETURN = false
```

```
Behavior = BHV_Waypoint
{
  name      = waypt_survey
  pwt       = 100
  condition = RETURN = false
  condition = DEPLOY = true
  endflag   = RETURN = true

           speed = 4
  capture_radius = 5.0
     slip_radius = 15.0
         polygon = 60,-40:60,-160:150,-160:180,-100:150,-40
         repeat  = 1
}
```

```
Behavior = BHV_Waypoint
{
  name       = waypt_return
  pwt        = 100
  condition  = RETURN = true
  condition  = DEPLOY = true
  endflag    = DEPLOY = false

           speed = 2.0
  capture_radius = 2.0
     slip_radius = 8.0
           point = 0,-2
}
```

The waypoint behavior accepts either:

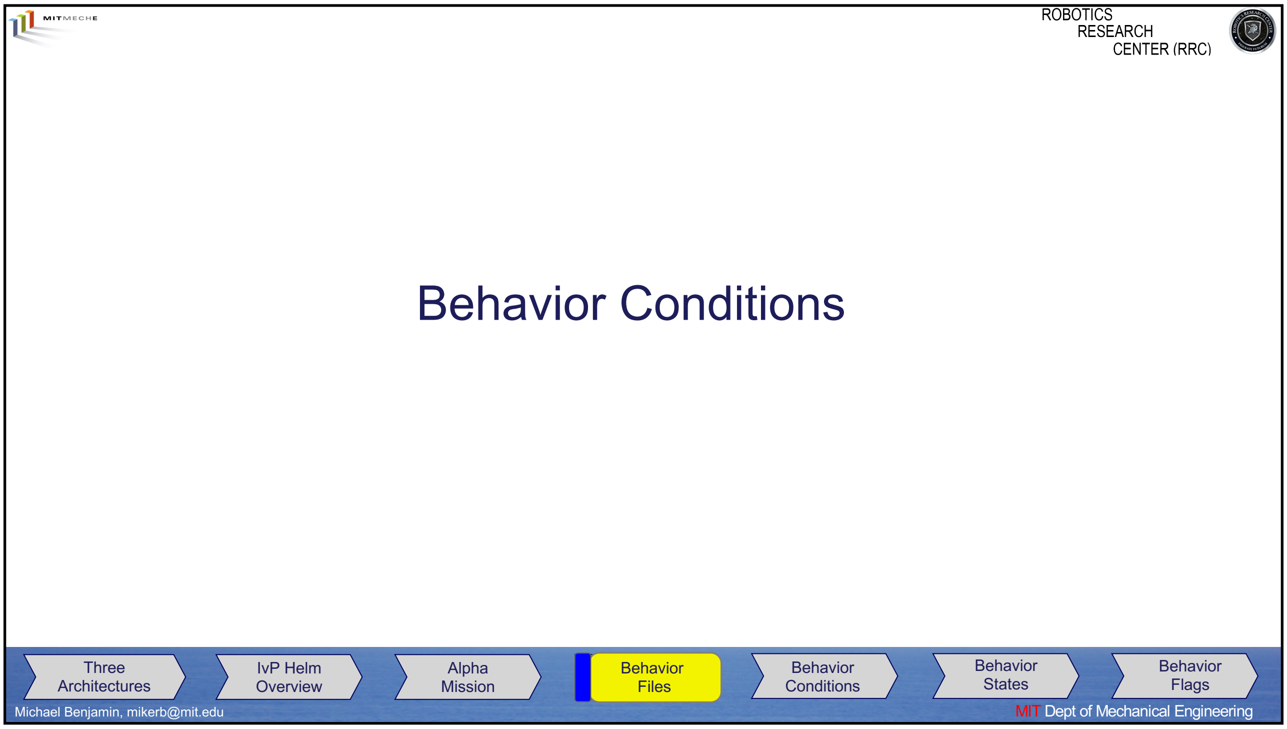
- *a polygon*
- *a single point*

Three Architectures | IvP Helm Overview | Alpha Mission | Behavior Files | Behavior Conditions | Behavior States | Behavior Flags

Michael Benjamin, mikerb@mit.edu

MIT Dept of Mechanical Engineering

32



ROBOTICS
RESEARCH
CENTER (RRC)

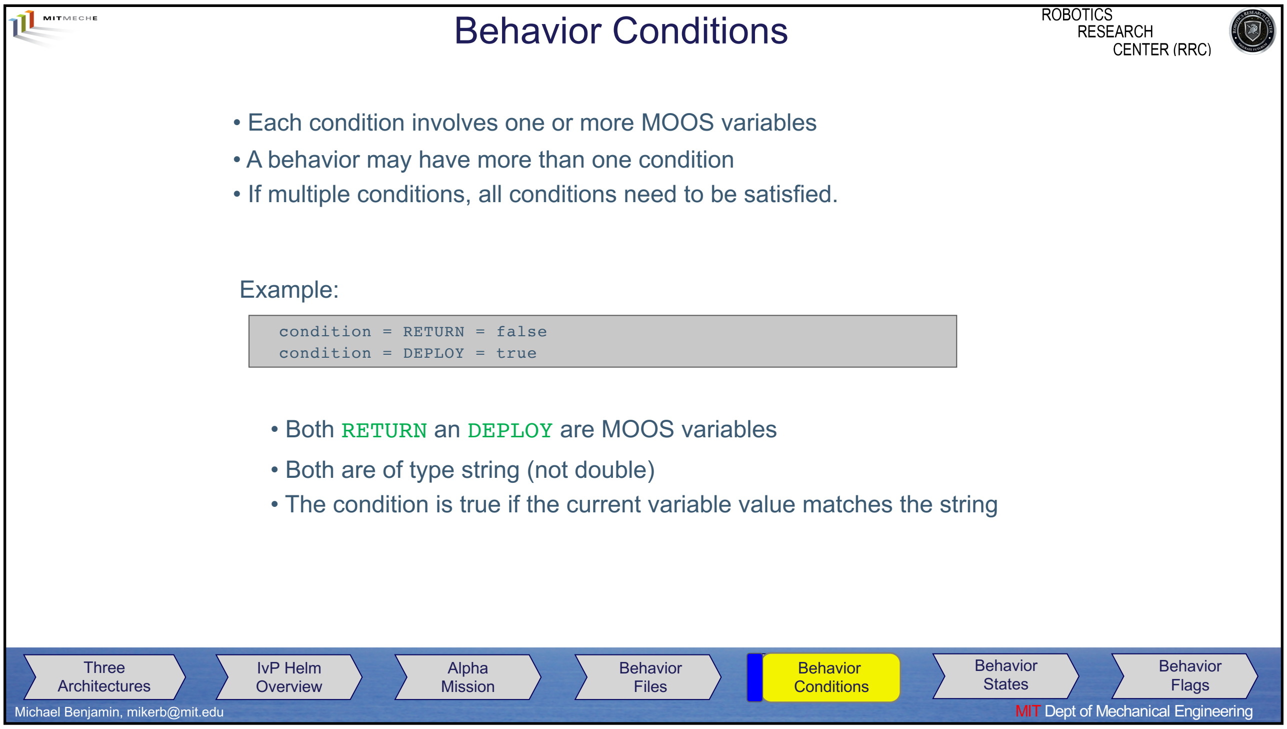
Behavior Conditions

Three Architectures
IvP Helm Overview
Alpha Mission
Behavior Files
Behavior Conditions
Behavior States
Behavior Flags

Michael Benjamin, mikerb@mit.edu

MIT Dept of Mechanical Engineering

33



ROBOTICS
RESEARCH
CENTER (RRC)

Behavior Conditions

- Each condition involves one or more MOOS variables
- A behavior may have more than one condition
- If multiple conditions, all conditions need to be satisfied.

Example:

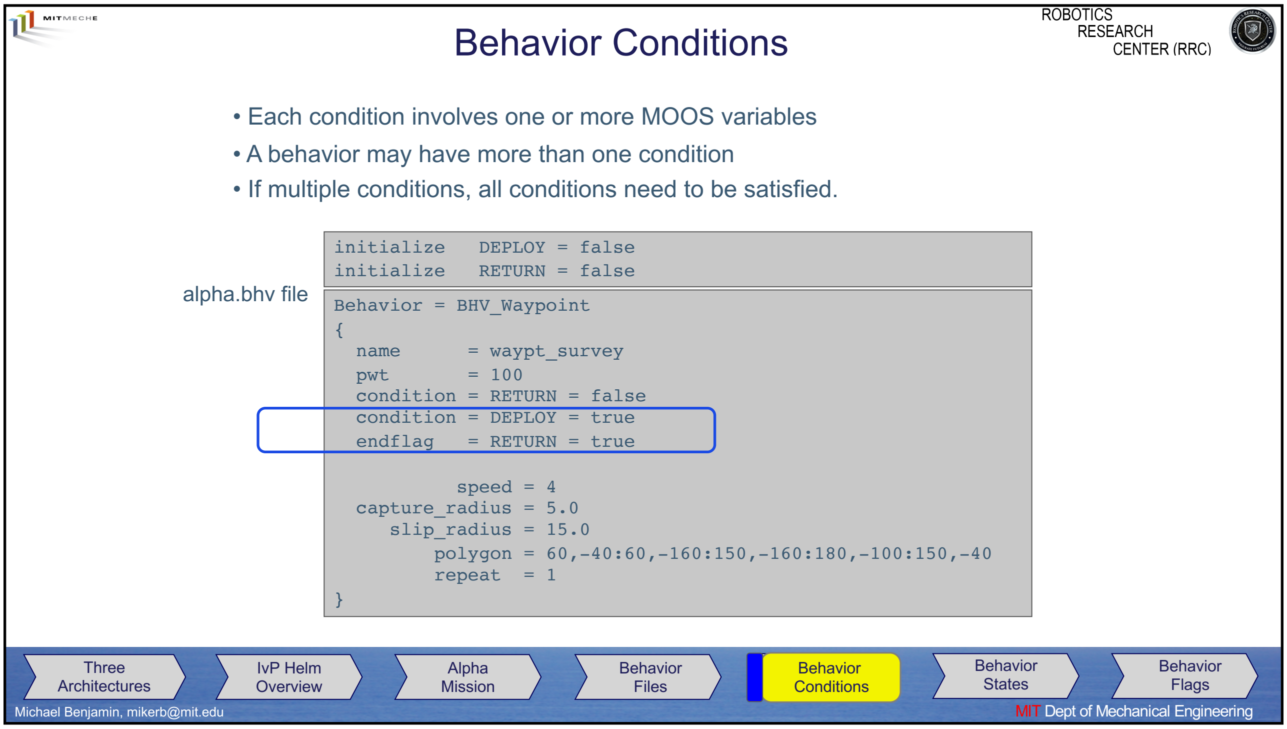
```
condition = RETURN = false
condition = DEPLOY = true
```

- Both `RETURN` an `DEPLOY` are MOOS variables
- Both are of type string (not double)
- The condition is true if the current variable value matches the string

Michael Benjamin, mikerb@mit.edu

MIT Dept of Mechanical Engineering

34



Behavior Conditions

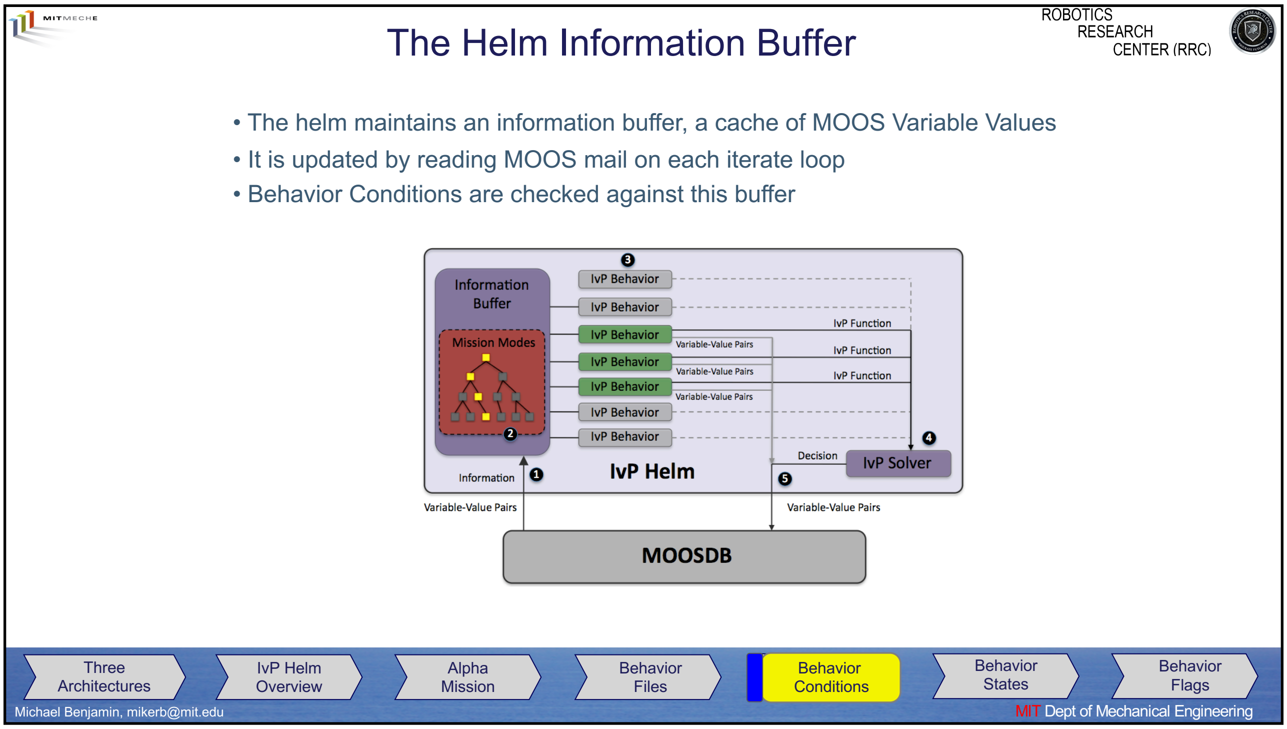
- Each condition involves one or more MOOS variables
- A behavior may have more than one condition
- If multiple conditions, all conditions need to be satisfied.

alpha.bhv file

```
initialize   DEPLOY = false
initialize   RETURN = false
```

```
Behavior = BHV_Waypoint
{
  name      = waypt_survey
  pwt       = 100
  condition = RETURN = false
  condition = DEPLOY = true
  endflag   = RETURN = true

           speed = 4
  capture_radius = 5.0
     slip_radius = 15.0
         polygon = 60,-40:60,-160:150,-160:180,-100:150,-40
         repeat  = 1
}
```

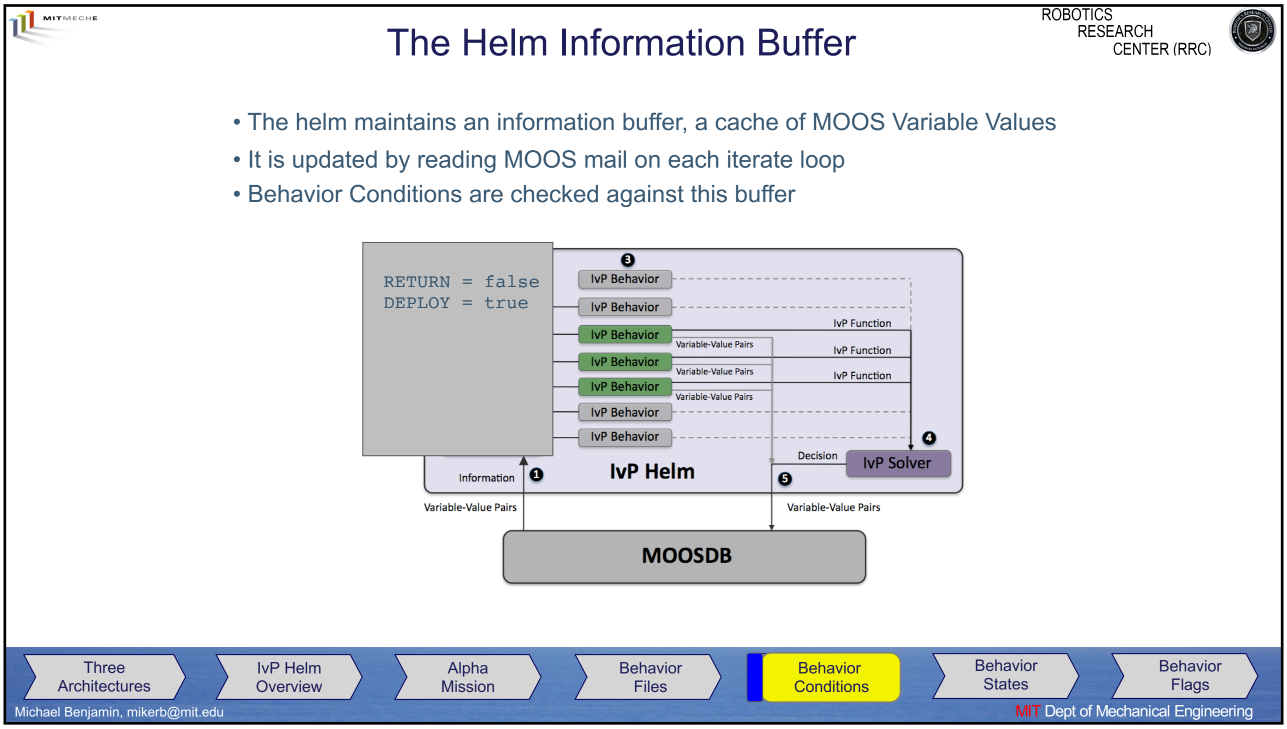
35



The Helm Information Buffer

- The helm maintains an information buffer, a cache of MOOS Variable Values
- It is updated by reading MOOS mail on each iterate loop
- Behavior Conditions are checked against this buffer

36



ROBOTICS
RESEARCH
CENTER (RRC)

The Helm Information Buffer

- The helm maintains an information buffer, a cache of MOOS Variable Values
- It is updated by reading MOOS mail on each iterate loop
- Behavior Conditions are checked against this buffer

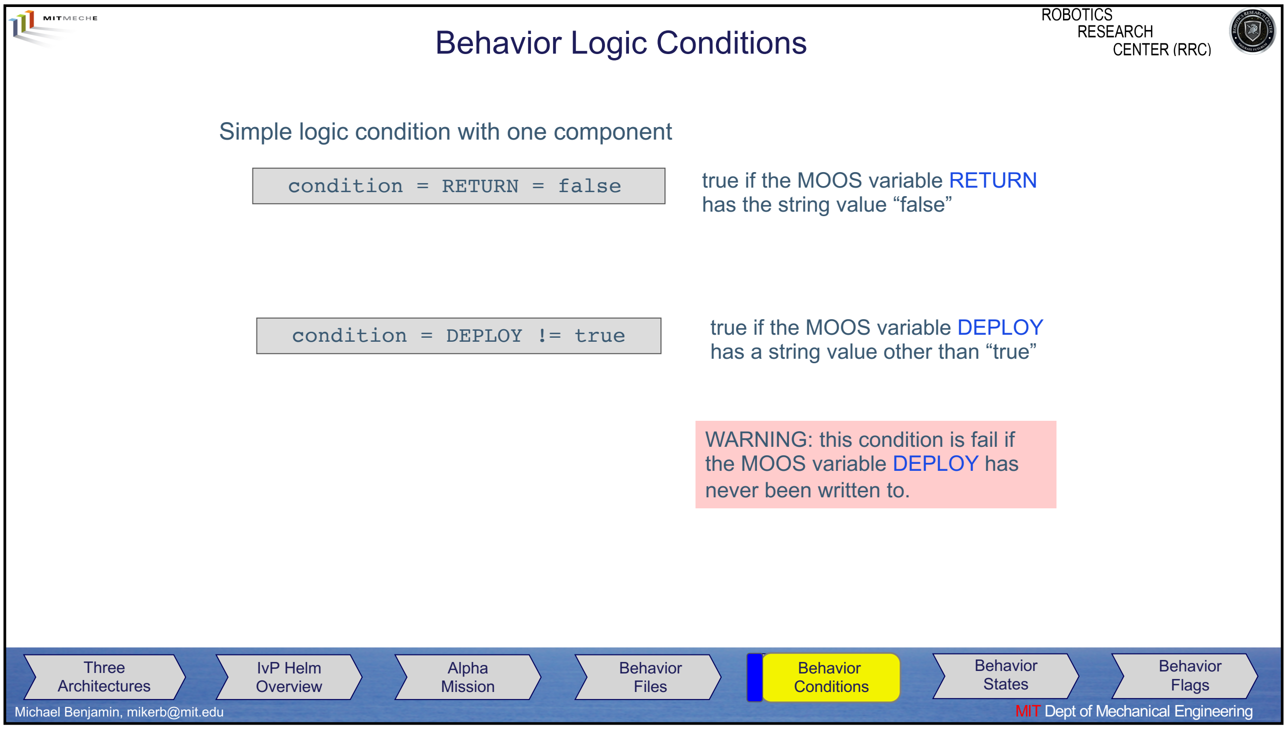
```
RETURN = false
DEPLOY = true
```

Three Architectures | IvP Helm Overview | Alpha Mission | Behavior Files | Behavior Conditions | Behavior States | Behavior Flags

Michael Benjamin, mikerb@mit.edu

MIT Dept of Mechanical Engineering

37



ROBOTICS
RESEARCH
CENTER (RRC)

Behavior Logic Conditions

Simple logic condition with one component

```
condition = RETURN = false
```

true if the MOOS variable RETURN has the string value “false”

```
condition = DEPLOY != true
```

true if the MOOS variable DEPLOY has a string value other than “true”

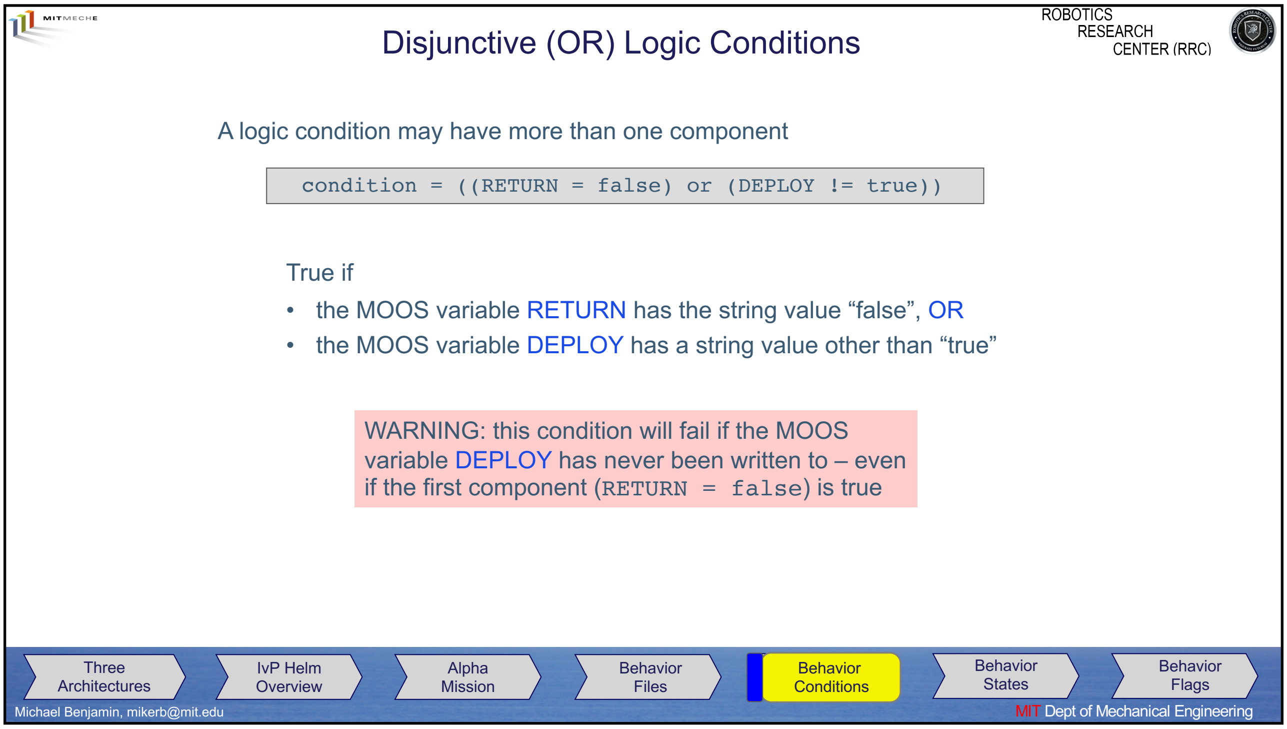
WARNING: this condition is fail if the MOOS variable DEPLOY has never been written to.

Three Architectures | IvP Helm Overview | Alpha Mission | Behavior Files | Behavior Conditions | Behavior States | Behavior Flags

Michael Benjamin, mikerb@mit.edu

MIT Dept of Mechanical Engineering

38



Disjunctive (OR) Logic Conditions

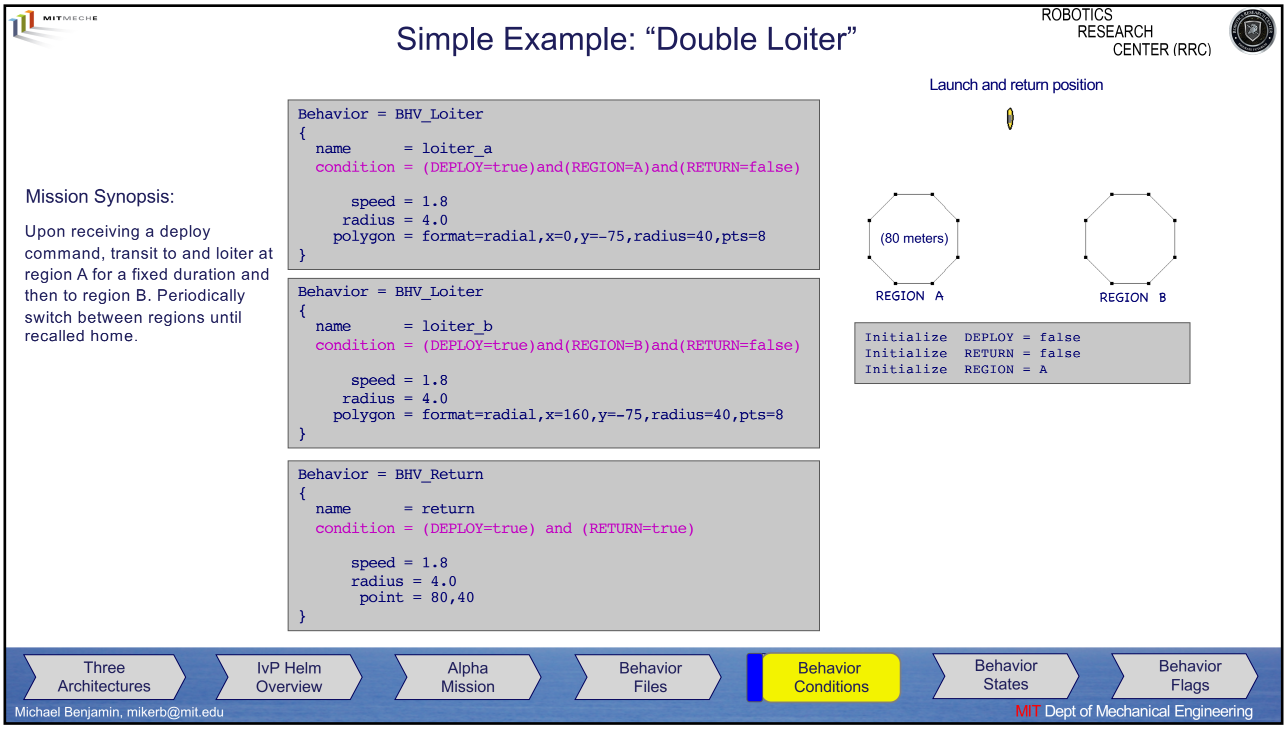
A logic condition may have more than one component

```
condition = ((RETURN = false) or (DEPLOY != true))
```

True if

- the MOOS variable RETURN has the string value “false”, OR
- the MOOS variable DEPLOY has a string value other than “true”

WARNING: this condition will fail if the MOOS variable DEPLOY has never been written to – even if the first component (`RETURN = false`) is true



Simple Example: “Double Loiter”

Mission Synopsis:

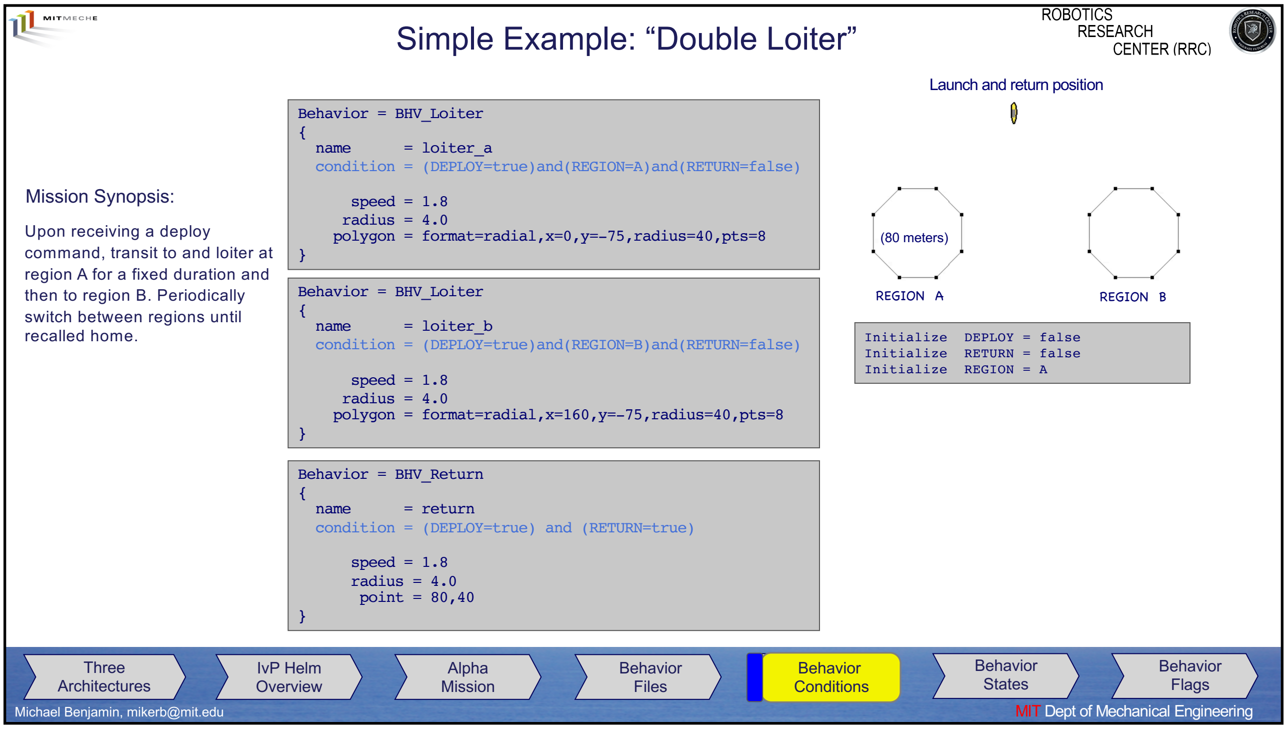
Upon receiving a deploy command, transit to and loiter at region A for a fixed duration and then to region B. Periodically switch between regions until recalled home.

```
Behavior = BHV_Loiter
{
  name      = loiter_a
  condition = (DEPLOY=true)and(REGION=A)and(RETURN=false)

      speed = 1.8
     radius = 4.0
    polygon = format=radial,x=0,y=-75,radius=40,pts=8
}
```

```
Behavior = BHV_Loiter
{
  name      = loiter_b
  condition = (DEPLOY=true)and(REGION=B)and(RETURN=false)

      speed = 1.8
     radius = 4.0
    polygon = format=radial,x=160,y=-75,radius=40,pts=8
}
```

```
Behavior = BHV_Return
{
  name      = return
  condition = (DEPLOY=true) and (RETURN=true)

      speed = 1.8
     radius = 4.0
      point = 80,40
}
```

```
Initialize  DEPLOY = false
Initialize  RETURN = false
Initialize  REGION = A
```



ROBOTICS
RESEARCH
CENTER (RRC)

Simple Example: “Double Loiter”

Mission Synopsis:

Upon receiving a deploy command, transit to and loiter at region A for a fixed duration and then to region B. Periodically switch between regions until recalled home.

```
Behavior = BHV_Loiter
{
  name      = loiter_a
  condition = (DEPLOY=true)and(REGION=A)and(RETURN=false)

      speed = 1.8
     radius = 4.0
    polygon = format=radial,x=0,y=-75,radius=40,pts=8
}
```

```
Behavior = BHV_Loiter
{
  name      = loiter_b
  condition = (DEPLOY=true)and(REGION=B)and(RETURN=false)

      speed = 1.8
     radius = 4.0
    polygon = format=radial,x=160,y=-75,radius=40,pts=8
}
```

```
Behavior = BHV_Return
{
  name      = return
  condition = (DEPLOY=true) and (RETURN=true)

      speed = 1.8
      radius = 4.0
       point = 80,40
}
```

Launch and return position

(80 meters)

REGION A

REGION B

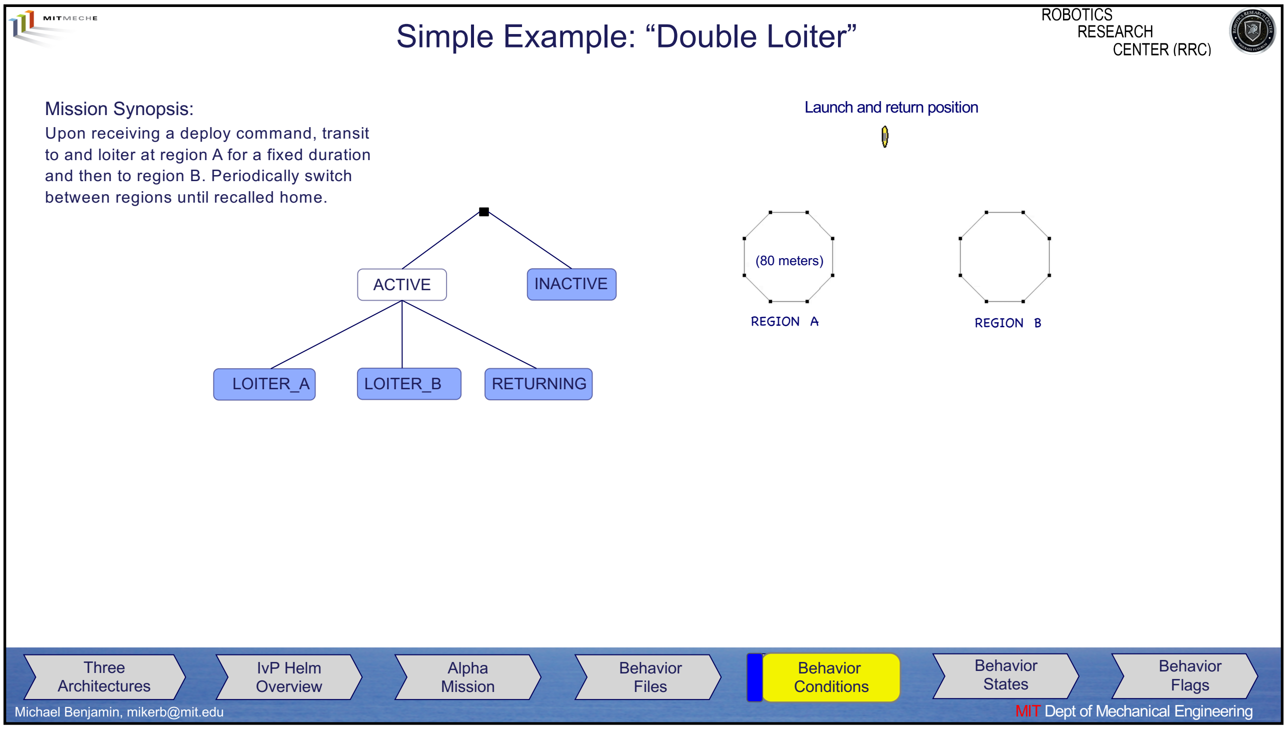
```
Initialize  DEPLOY = false
Initialize  RETURN = false
Initialize  REGION = A
```

Three Architectures | IvP Helm Overview | Alpha Mission | Behavior Files | Behavior Conditions | Behavior States | Behavior Flags

Michael Benjamin, mikerb@mit.edu

MIT Dept of Mechanical Engineering

41



ROBOTICS
RESEARCH
CENTER (RRC)

Simple Example: “Double Loiter”

Mission Synopsis:

Upon receiving a deploy command, transit to and loiter at region A for a fixed duration and then to region B. Periodically switch between regions until recalled home.

ACTIVE

INACTIVE

LOITER_A

LOITER_B

RETURNING

Launch and return position

(80 meters)

REGION A

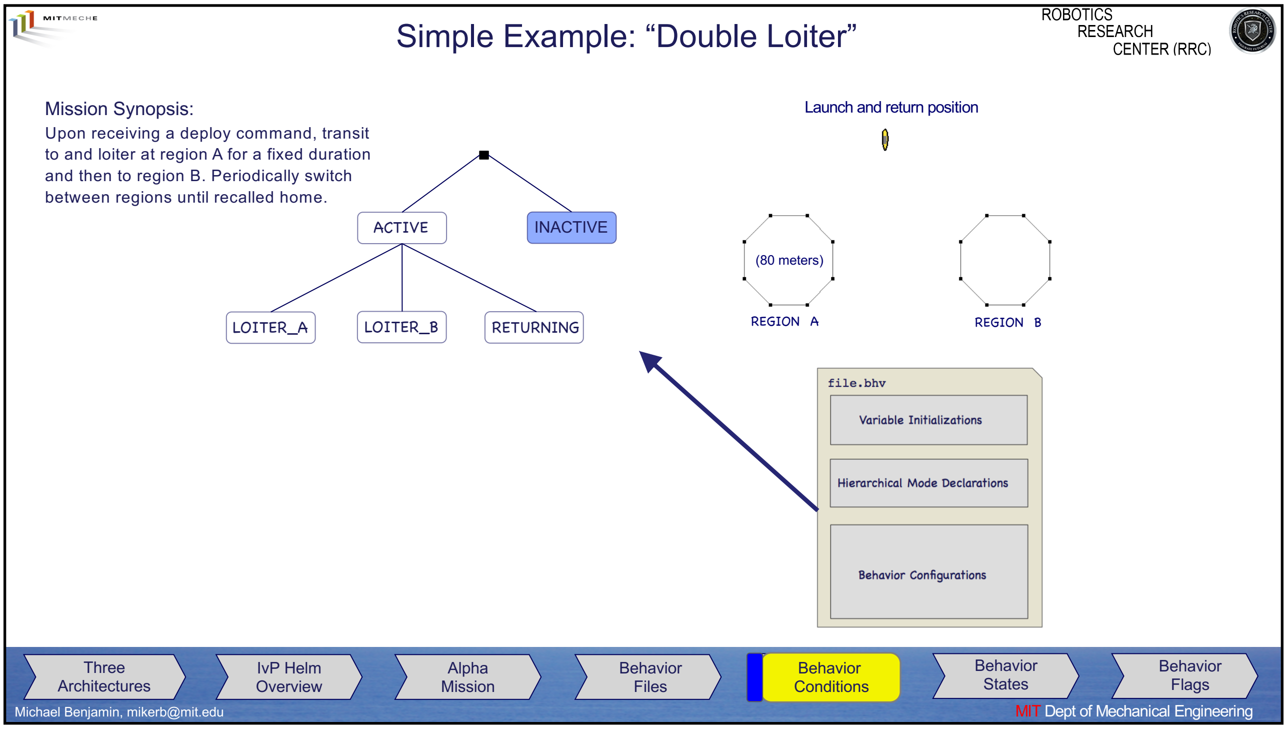
REGION B

Three Architectures | IvP Helm Overview | Alpha Mission | Behavior Files | Behavior Conditions | Behavior States | Behavior Flags

Michael Benjamin, mikerb@mit.edu

MIT Dept of Mechanical Engineering

42



ROBOTICS RESEARCH CENTER (RRC)

Simple Example: "Double Loiter"

Mission Synopsis:

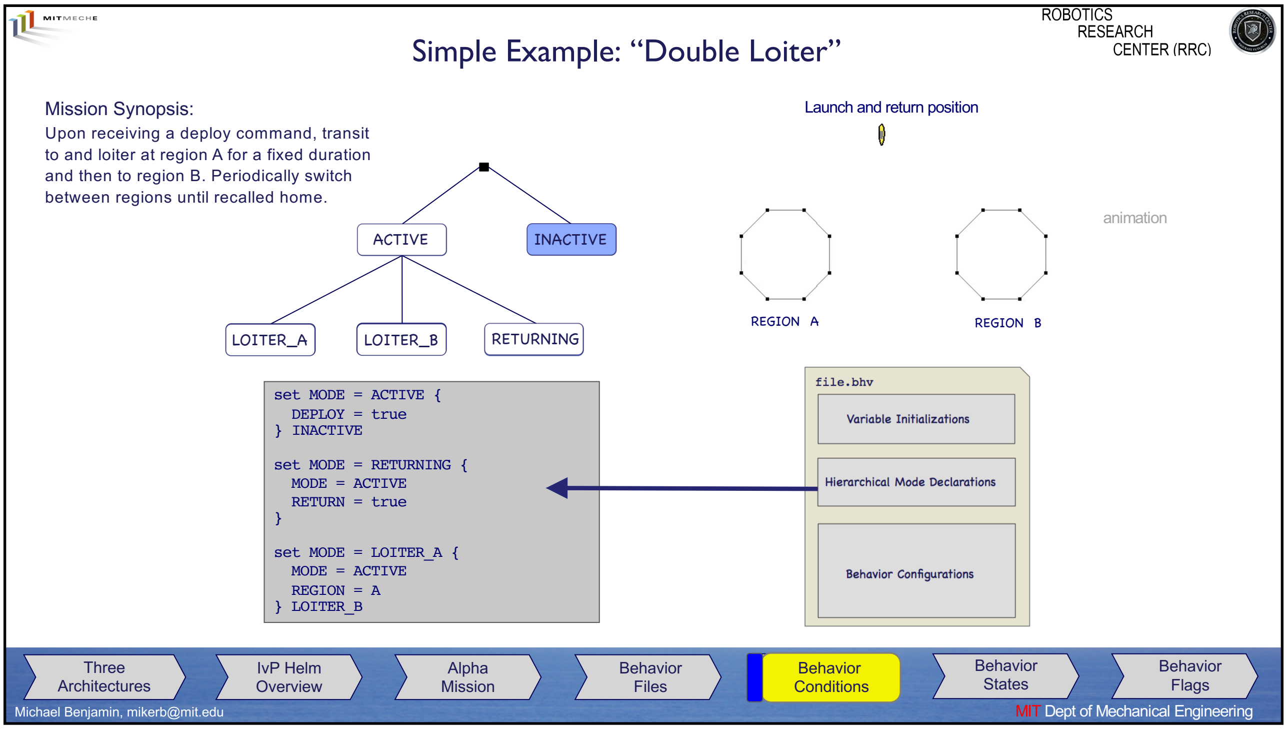
Upon receiving a deploy command, transit to and loiter at region A for a fixed duration and then to region B. Periodically switch between regions until recalled home.

Three Architectures | IvP Helm Overview | Alpha Mission | Behavior Files | Behavior Conditions | Behavior States | Behavior Flags

Michael Benjamin, mikerb@mit.edu

MIT Dept of Mechanical Engineering

43



ROBOTICS RESEARCH CENTER (RRC)

Simple Example: "Double Loiter"

Mission Synopsis:

Upon receiving a deploy command, transit to and loiter at region A for a fixed duration and then to region B. Periodically switch between regions until recalled home.

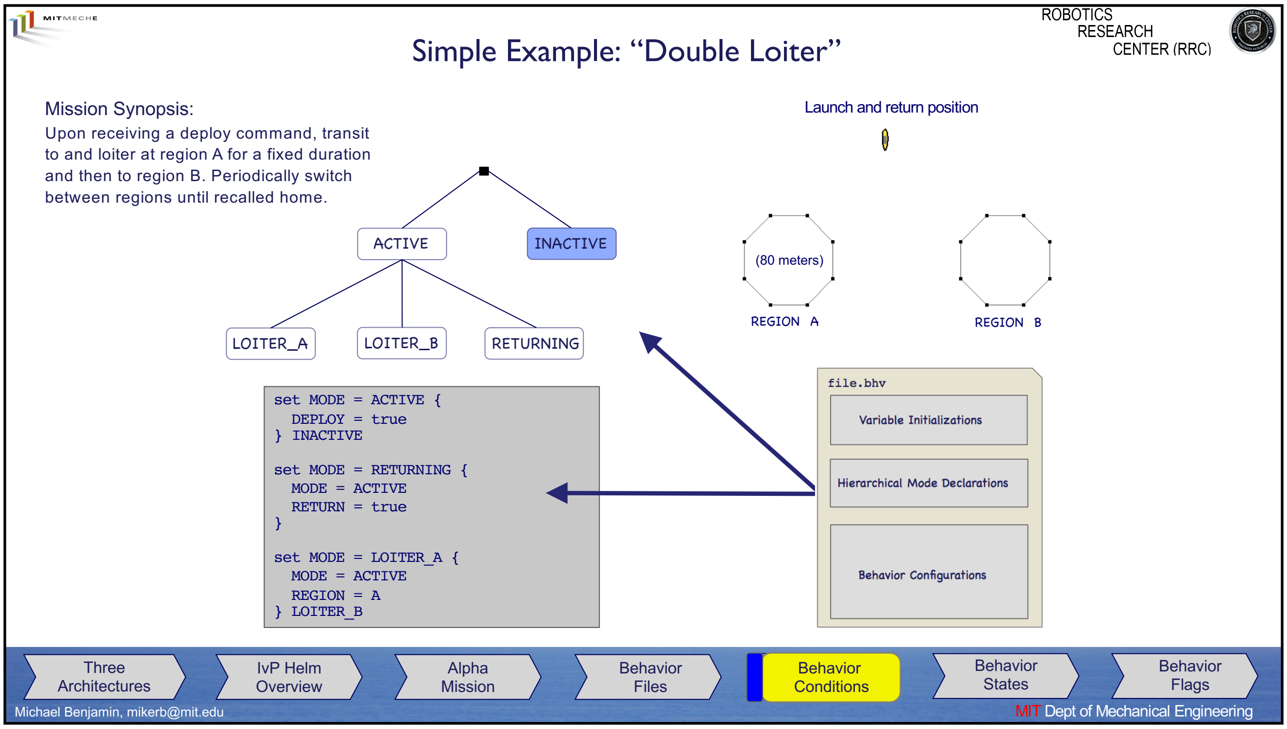
```
set MODE = ACTIVE {
  DEPLOY = true
} INACTIVE

set MODE = RETURNING {
  MODE = ACTIVE
  RETURN = true
}

set MODE = LOITER_A {
  MODE = ACTIVE
  REGION = A
} LOITER_B
```

Three Architectures | IvP Helm Overview | Alpha Mission | Behavior Files | Behavior Conditions | Behavior States | Behavior Flags

Michael Benjamin, mikerb@mit.edu

MIT Dept of Mechanical Engineering

44



ROBOTICS RESEARCH CENTER (RRC)

Simple Example: “Double Loiter”

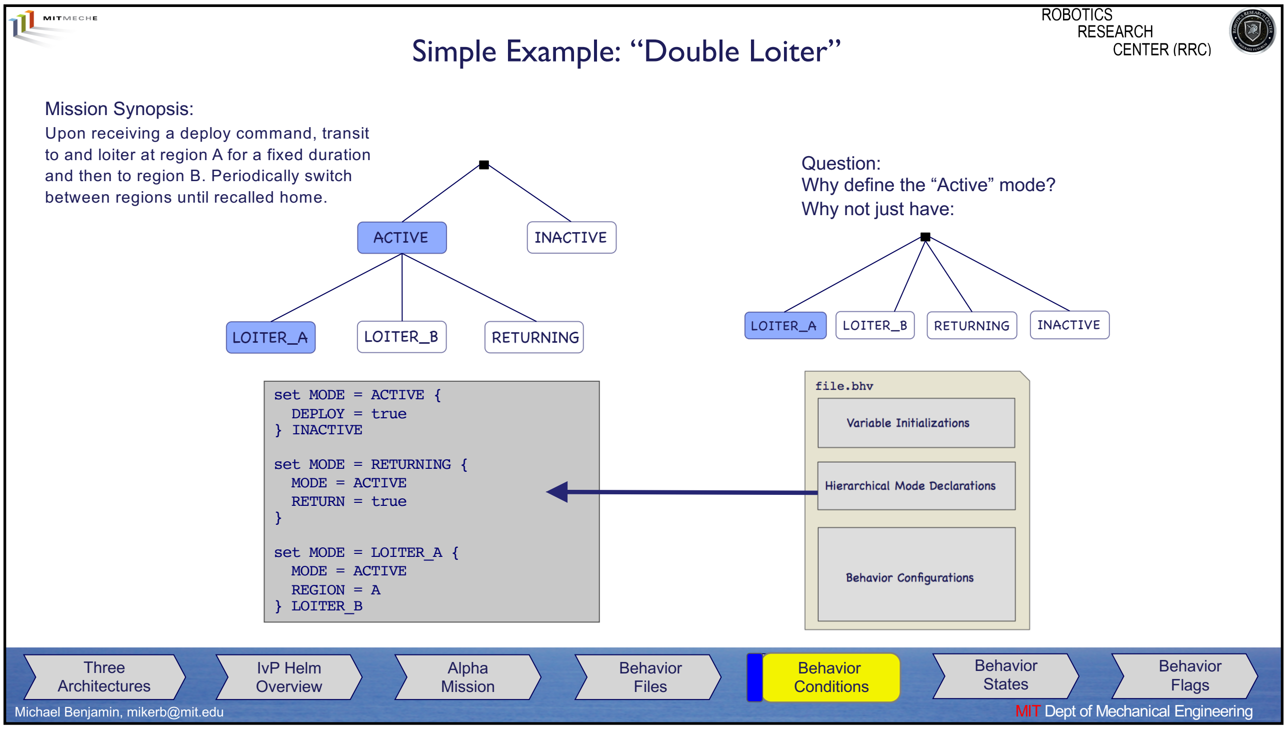
Mission Synopsis:
Upon receiving a deploy command, transit to and loiter at region A for a fixed duration and then to region B. Periodically switch between regions until recalled home.

```
set MODE = ACTIVE {
  DEPLOY = true
} INACTIVE

set MODE = RETURNING {
  MODE = ACTIVE
  RETURN = true
}

set MODE = LOITER_A {
  MODE = ACTIVE
  REGION = A
} LOITER_B
```

Michael Benjamin, mikerb@mit.edu

MIT Dept of Mechanical Engineering

45



ROBOTICS RESEARCH CENTER (RRC)

Simple Example: “Double Loiter”

Mission Synopsis:
Upon receiving a deploy command, transit to and loiter at region A for a fixed duration and then to region B. Periodically switch between regions until recalled home.

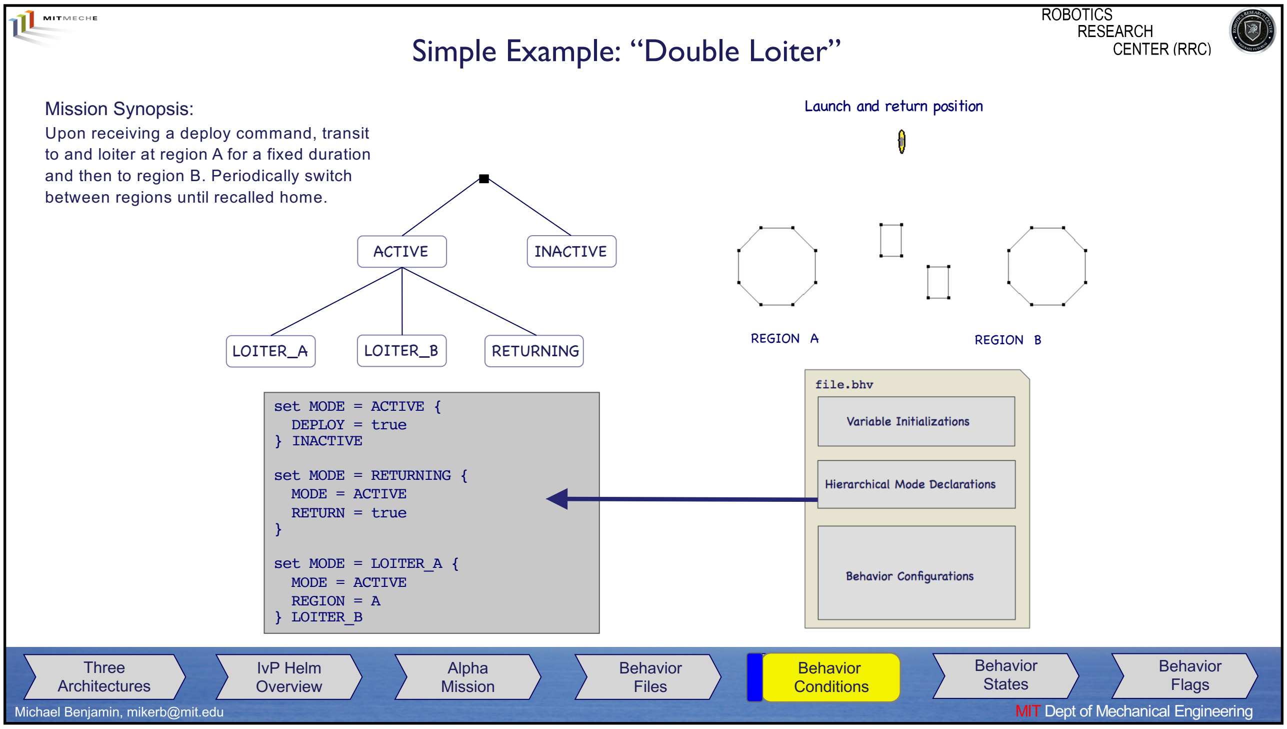
Question:
Why define the “Active” mode?
Why not just have:

```
set MODE = ACTIVE {
  DEPLOY = true
} INACTIVE

set MODE = RETURNING {
  MODE = ACTIVE
  RETURN = true
}

set MODE = LOITER_A {
  MODE = ACTIVE
  REGION = A
} LOITER_B
```

Michael Benjamin, mikerb@mit.edu

MIT Dept of Mechanical Engineering

46



MIT MECHE

ROBOTICS
RESEARCH
CENTER (RRC)

Simple Example: “Double Loiter”

Mission Synopsis:

Upon receiving a deploy command, transit to and loiter at region A for a fixed duration and then to region B. Periodically switch between regions until recalled home.

ACTIVE

INACTIVE

LOITER_A

LOITER_B

RETURNING

```
set MODE = ACTIVE {
  DEPLOY = true
} INACTIVE

set MODE = RETURNING {
  MODE = ACTIVE
  RETURN = true
}

set MODE = LOITER_A {
  MODE = ACTIVE
  REGION = A
} LOITER_B
```

Launch and return position

REGION A

REGION B

file.bhv

Variable Initializations

Hierarchical Mode Declarations

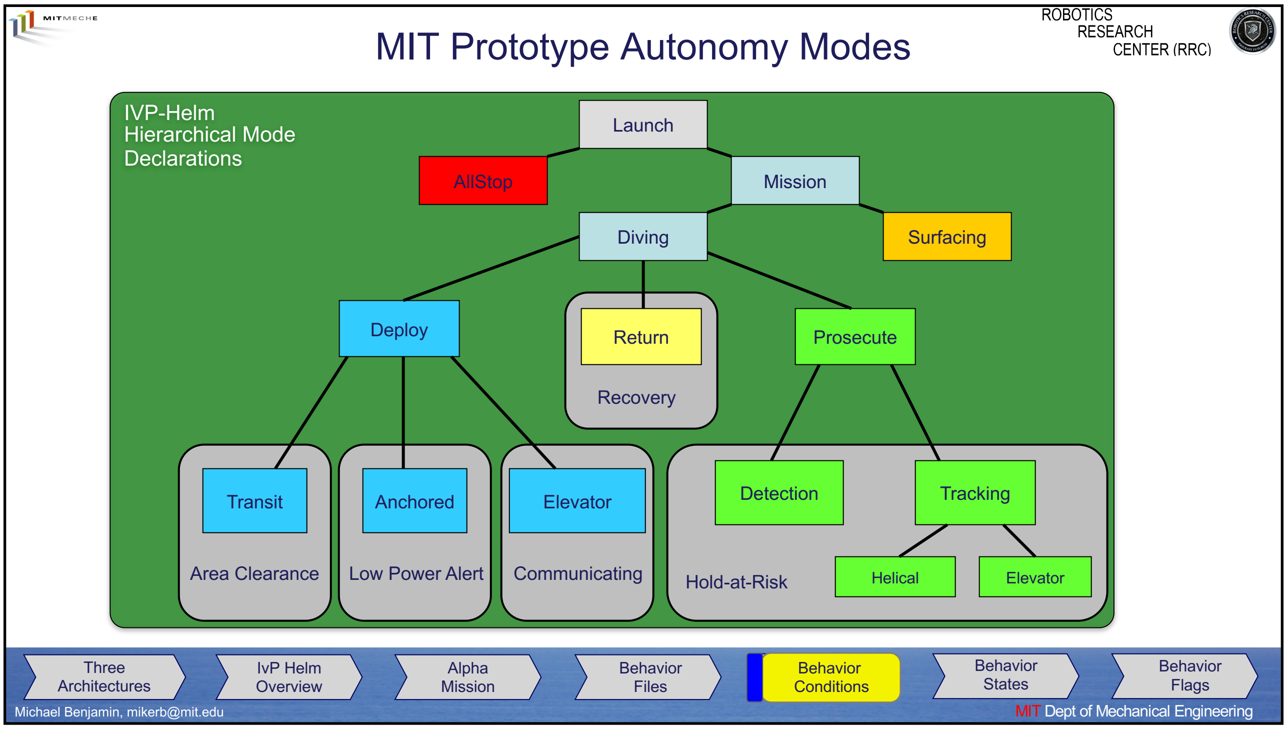
Behavior Configurations

Three Architectures | IvP Helm Overview | Alpha Mission | Behavior Files | Behavior Conditions | Behavior States | Behavior Flags

Michael Benjamin, mikerb@mit.edu

MIT Dept of Mechanical Engineering

47



MIT MECHE

ROBOTICS
RESEARCH
CENTER (RRC)

MIT Prototype Autonomy Modes

IVP-Helm
Hierarchical Mode
Declarations

Launch

AllStop

Mission

Diving

Surfacing

Deploy

Return

Recovery

Prosecute

Transit

Area Clearance

Anchored

Low Power Alert

Elevator

Communicating

Detection

Tracking

Hold-at-Risk

Helical

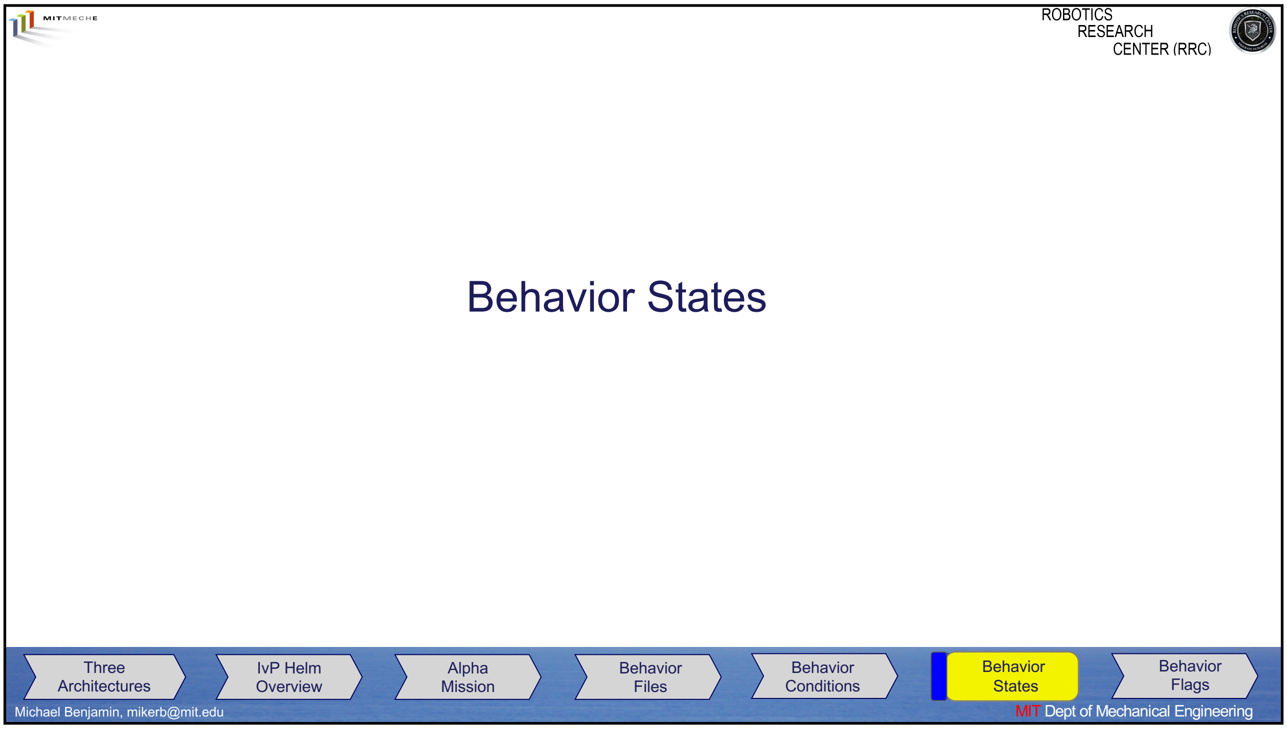
Elevator

Three Architectures | IvP Helm Overview | Alpha Mission | Behavior Files | Behavior Conditions | Behavior States | Behavior Flags

Michael Benjamin, mikerb@mit.edu

MIT Dept of Mechanical Engineering

48



MITMECHE

ROBOTICS
RESEARCH
CENTER (RRC)

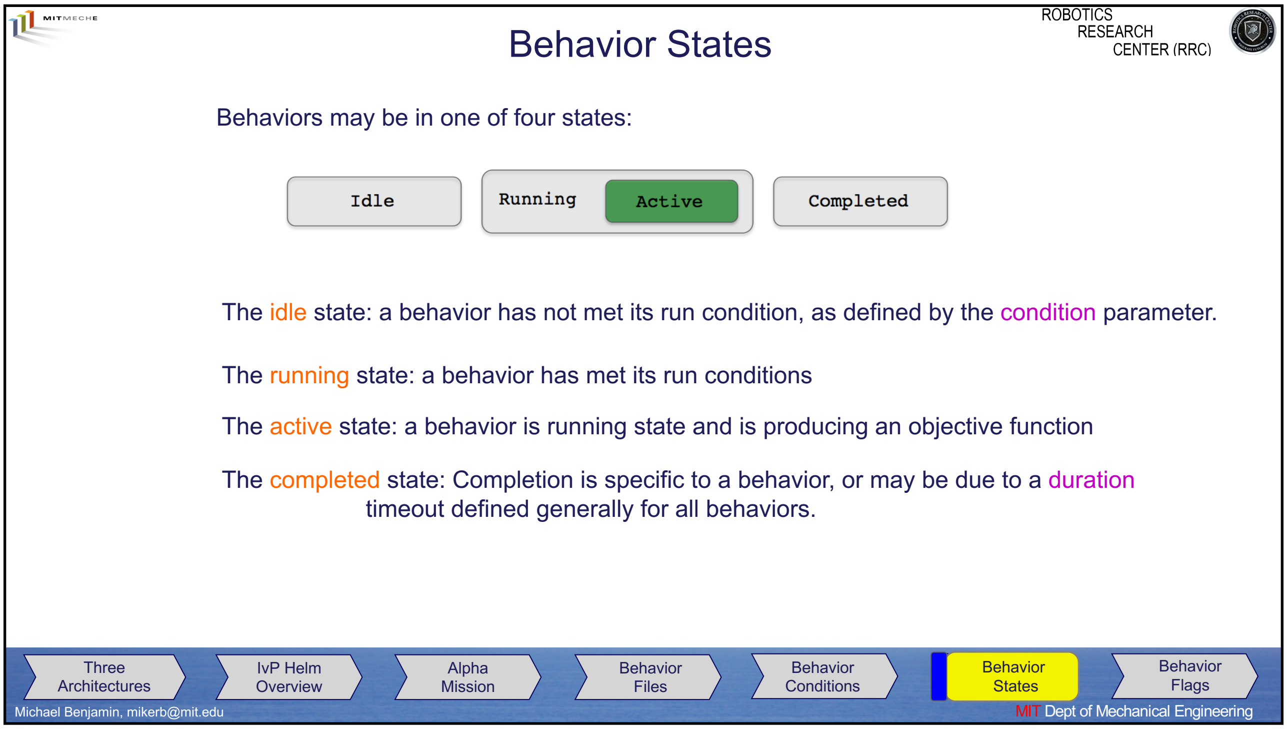
Behavior States

Three Architectures | IvP Helm Overview | Alpha Mission | Behavior Files | Behavior Conditions | Behavior States | Behavior Flags

Michael Benjamin, mikerb@mit.edu

MIT Dept of Mechanical Engineering

49



MITMECHE

ROBOTICS
RESEARCH
CENTER (RRC)

Behavior States

Behaviors may be in one of four states:

`Idle` `Running` `Active` `Completed`

The idle state: a behavior has not met its run condition, as defined by the condition parameter.

The running state: a behavior has met its run conditions

The active state: a behavior is running state and is producing an objective function

The completed state: Completion is specific to a behavior, or may be due to a duration timeout defined generally for all behaviors.

Three Architectures | IvP Helm Overview | Alpha Mission | Behavior Files | Behavior Conditions | Behavior States | Behavior Flags

Michael Benjamin, mikerb@mit.edu

MIT Dept of Mechanical Engineering

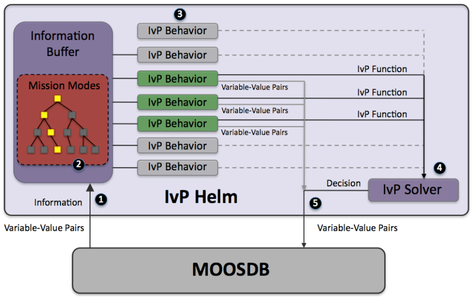
50

MITMECHE

ROBOTICS
RESEARCH
CENTER (RRC)

Active vs. Running States

Idle | Running | Active | Completed



The running state: behavior has met its run conditions.
The active state: behavior is running and producing an objective function.

The helm's primary job is to produce a helm decision. A behavior is participating in that decision only if it is producing an objective function.

A behavior may participate in the helm decision based on:

(1) The run conditions (mostly dependent on an external decision process)

(2) The behavior's own logic (a local decision based on a more nuanced understanding of the situation).

Three Architectures | IvP Helm Overview | Alpha Mission | Behavior Files | Behavior Conditions | Behavior States | Behavior Flags

Michael Benjamin, mikerb@mit.edu

MIT Dept of Mechanical Engineering

51

MITMECHE

ROBOTICS
RESEARCH
CENTER (RRC)

Active vs. Running States

What is the difference between running and active?

Idle | Running | Active | Completed

The running state: behavior has met its run conditions.
The active state: behavior is running and producing an objective function.

The helm's primary job is to produce a helm decision. A behavior is participating in that decision only if it is producing an objective function.

The choice to participate in the helm decision is made at two points:

(1) The run conditions (mostly dependent on an external decision process)

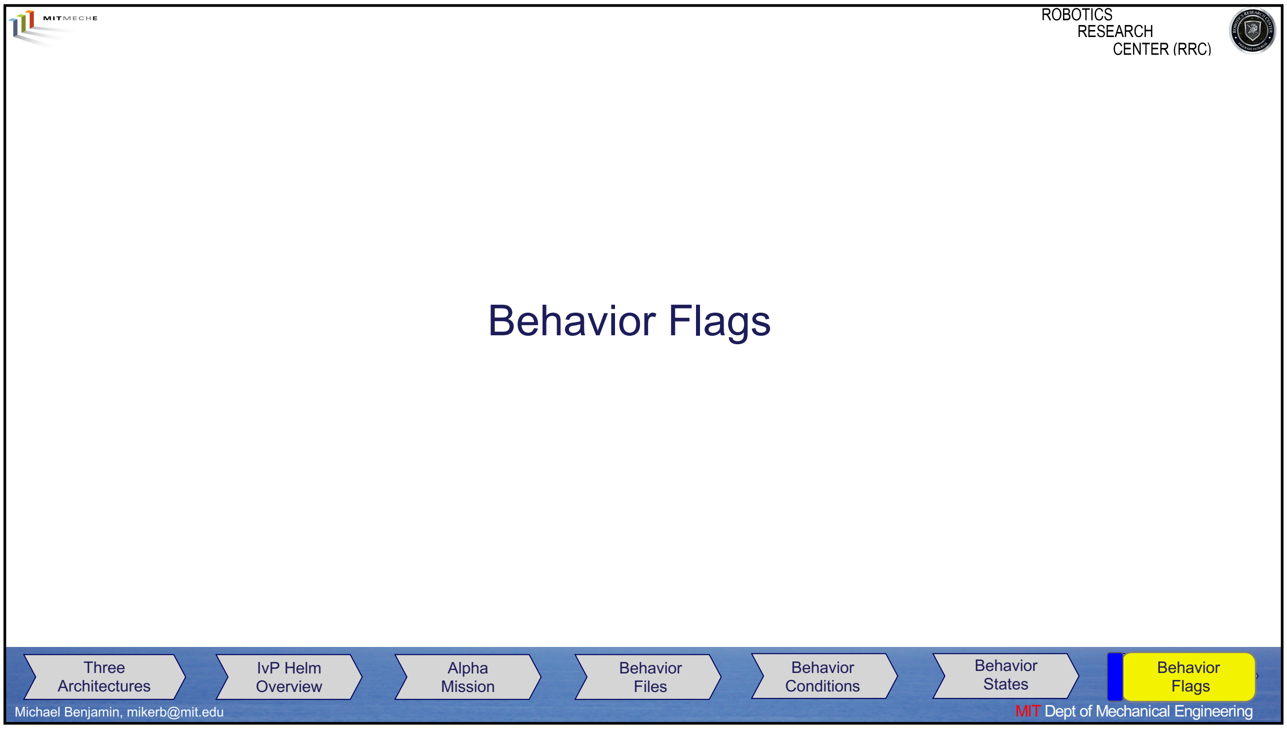
(2) The behavior's own logic (a local decision based on a more nuanced / domain-expert understanding of the situation).

Three Architectures | IvP Helm Overview | Alpha Mission | Behavior Files | Behavior Conditions | Behavior States | Behavior Flags

Michael Benjamin, mikerb@mit.edu

MIT Dept of Mechanical Engineering

52



ROBOTICS
RESEARCH
CENTER (RRC)

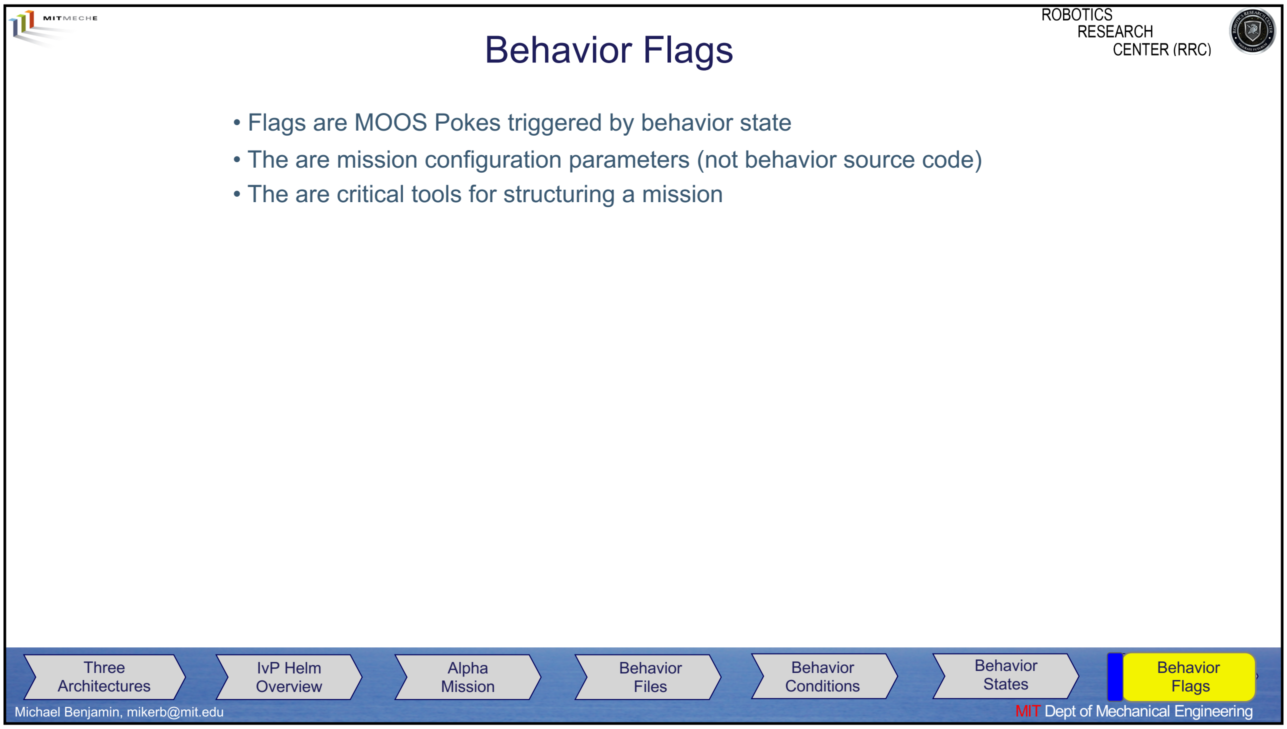
Behavior Flags

Three Architectures | IvP Helm Overview | Alpha Mission | Behavior Files | Behavior Conditions | Behavior States | Behavior Flags

Michael Benjamin, mikerb@mit.edu

MIT Dept of Mechanical Engineering

53



ROBOTICS
RESEARCH
CENTER (RRC)

Behavior Flags

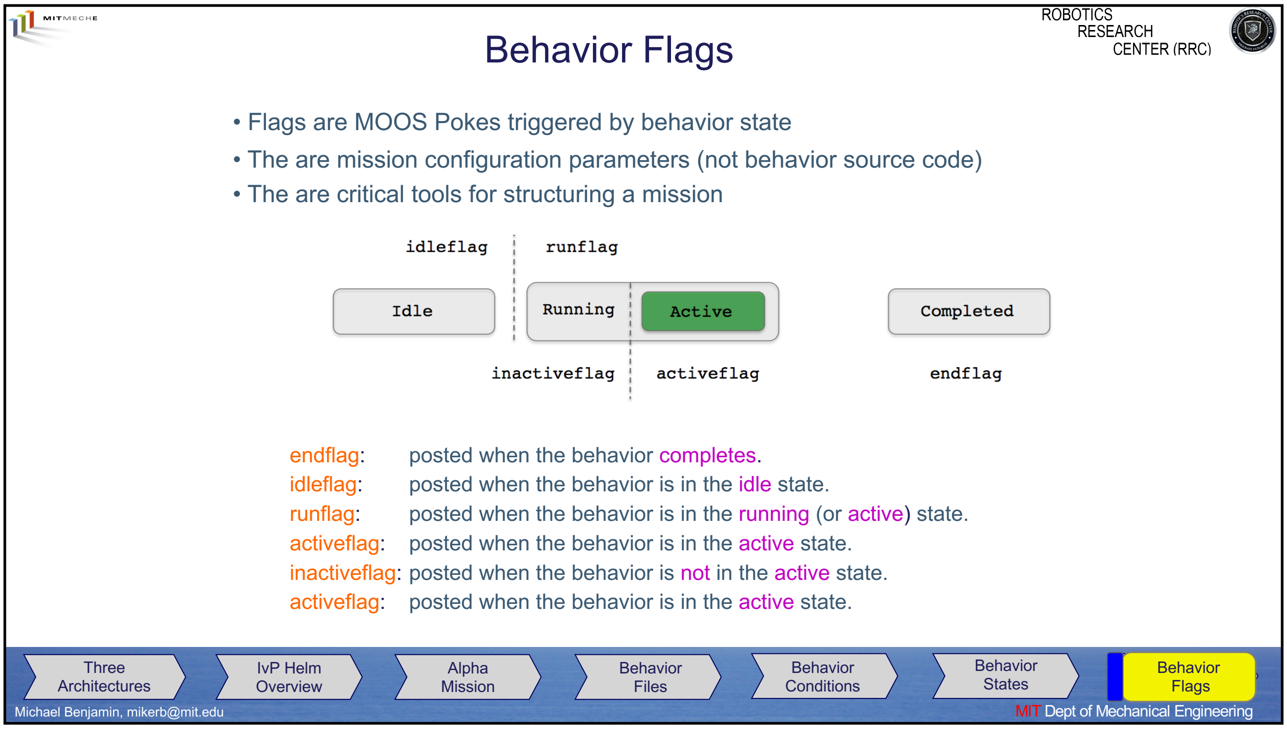
- Flags are MOOS Pokes triggered by behavior state
- The are mission configuration parameters (not behavior source code)
- The are critical tools for structuring a mission

Three Architectures | IvP Helm Overview | Alpha Mission | Behavior Files | Behavior Conditions | Behavior States | Behavior Flags

Michael Benjamin, mikerb@mit.edu

MIT Dept of Mechanical Engineering

54



ROBOTICS RESEARCH CENTER (RRC)

Behavior Flags

- Flags are MOOS Pokes triggered by behavior state
- The are mission configuration parameters (not behavior source code)
- The are critical tools for structuring a mission

idleflag

runflag

Idle

Running

Active

Completed

inactiveflag

activeflag

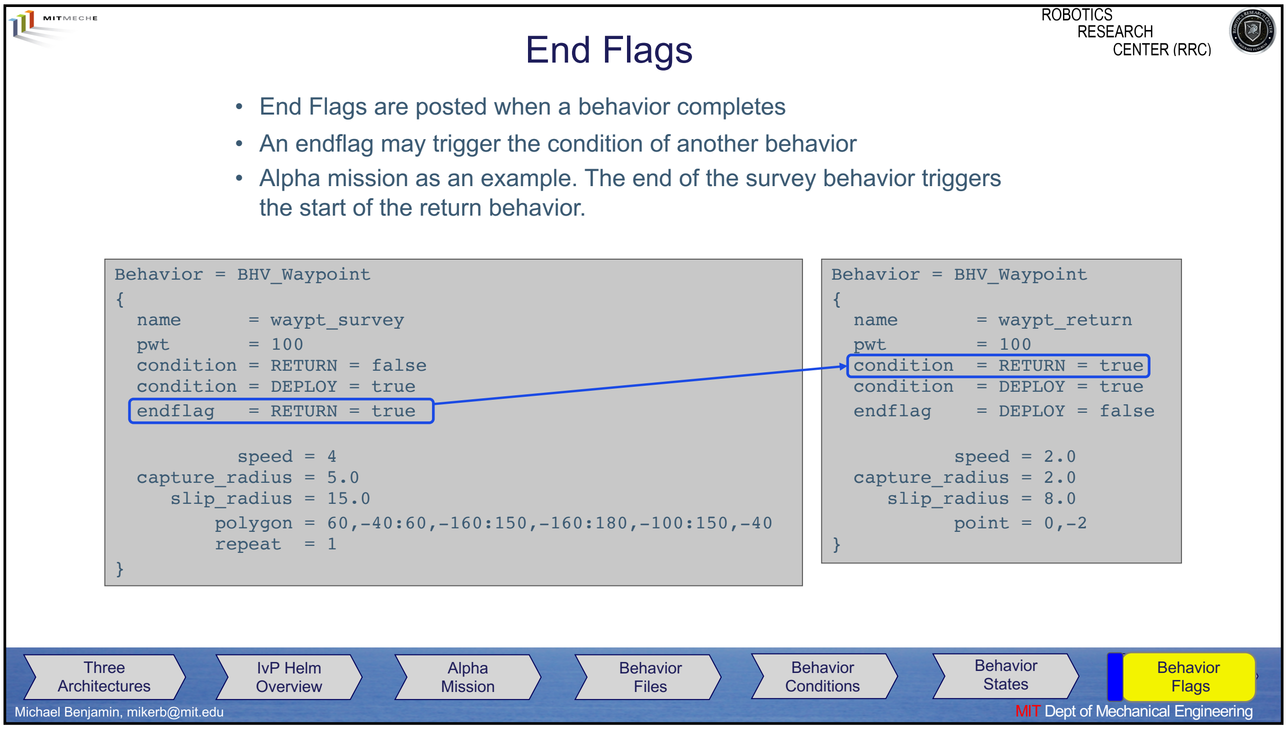
endflag

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 (or active) 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.

Michael Benjamin, mikerb@mit.edu

MIT Dept of Mechanical Engineering

55



ROBOTICS RESEARCH CENTER (RRC)

End Flags

- End Flags are posted when a behavior completes
- An endflag may trigger the condition of another behavior
- Alpha mission as an example. The end of the survey behavior triggers the start of the return behavior.

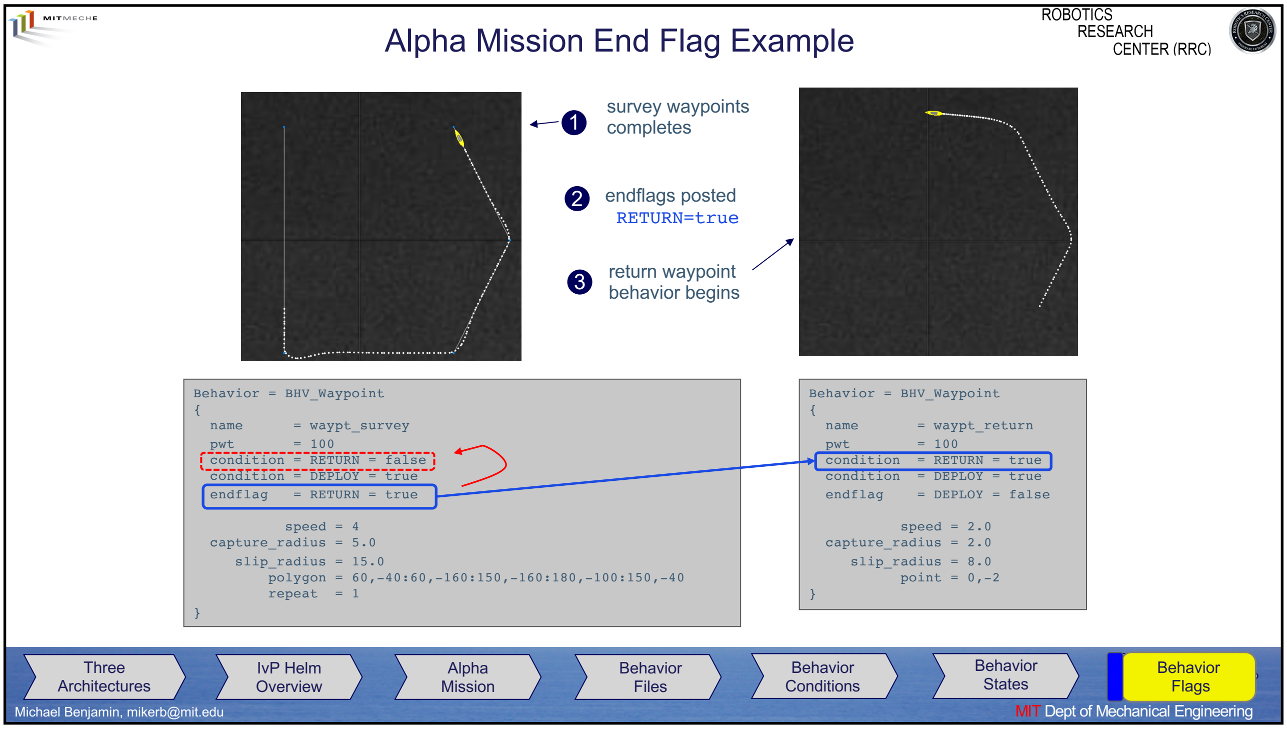
```
Behavior = BHV_Waypoint
{
  name      = waypt_survey
  pwt       = 100
  condition = RETURN = false
  condition = DEPLOY = true
  endflag   = RETURN = true

         speed = 4
  capture_radius = 5.0
     slip_radius = 15.0
         polygon = 60,-40:60,-160:150,-160:180,-100:150,-40
         repeat  = 1
}
```

```
Behavior = BHV_Waypoint
{
  name        = waypt_return
  pwt         = 100
  condition   = RETURN = true
  condition   = DEPLOY = true
  endflag     = DEPLOY = false

          speed = 2.0
  capture_radius = 2.0
     slip_radius = 8.0
           point = 0,-2
}
```

Michael Benjamin, mikerb@mit.edu

MIT Dept of Mechanical Engineering

56



ROBOTICS RESEARCH CENTER (RRC)

Alpha Mission End Flag Example

1 survey waypoints completes

2 endflags posted
RETURN=true

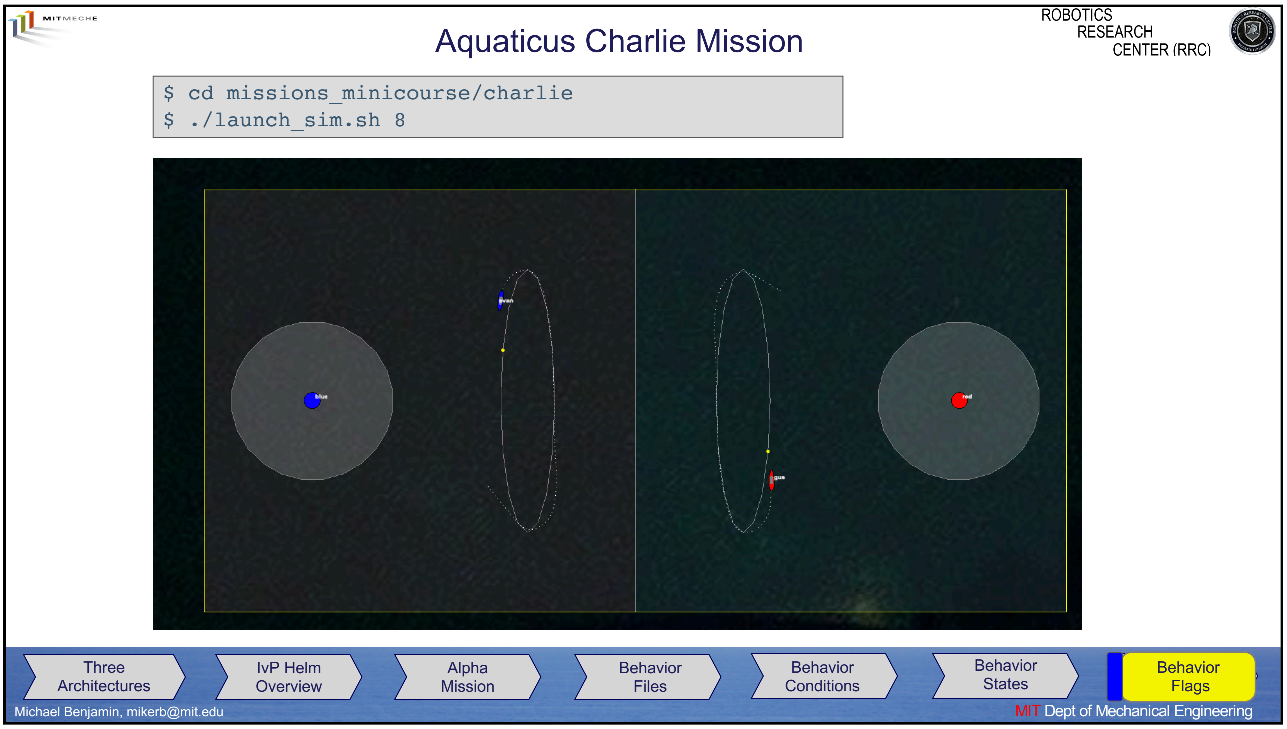
3 return waypoint behavior begins

```
Behavior = BHV_Waypoint
{
  name       = waypt_survey
  pwt        = 100
  condition = RETURN = false
  condition = DEPLOY = true
  endflag    = RETURN = true

           speed = 4
  capture_radius = 5.0
     slip_radius = 15.0
         polygon = 60,-40:60,-160:150,-160:180,-100:150,-40
         repeat  = 1
}
```

```
Behavior = BHV_Waypoint
{
  name        = waypt_return
  pwt         = 100
  condition   = RETURN = true
  condition   = DEPLOY = true
  endflag     = DEPLOY = false

           speed = 2.0
  capture_radius = 2.0
     slip_radius = 8.0
           point = 0,-2
}
```

Three Architectures | IvP Helm Overview | Alpha Mission | Behavior Files | Behavior Conditions | Behavior States | Behavior Flags

57



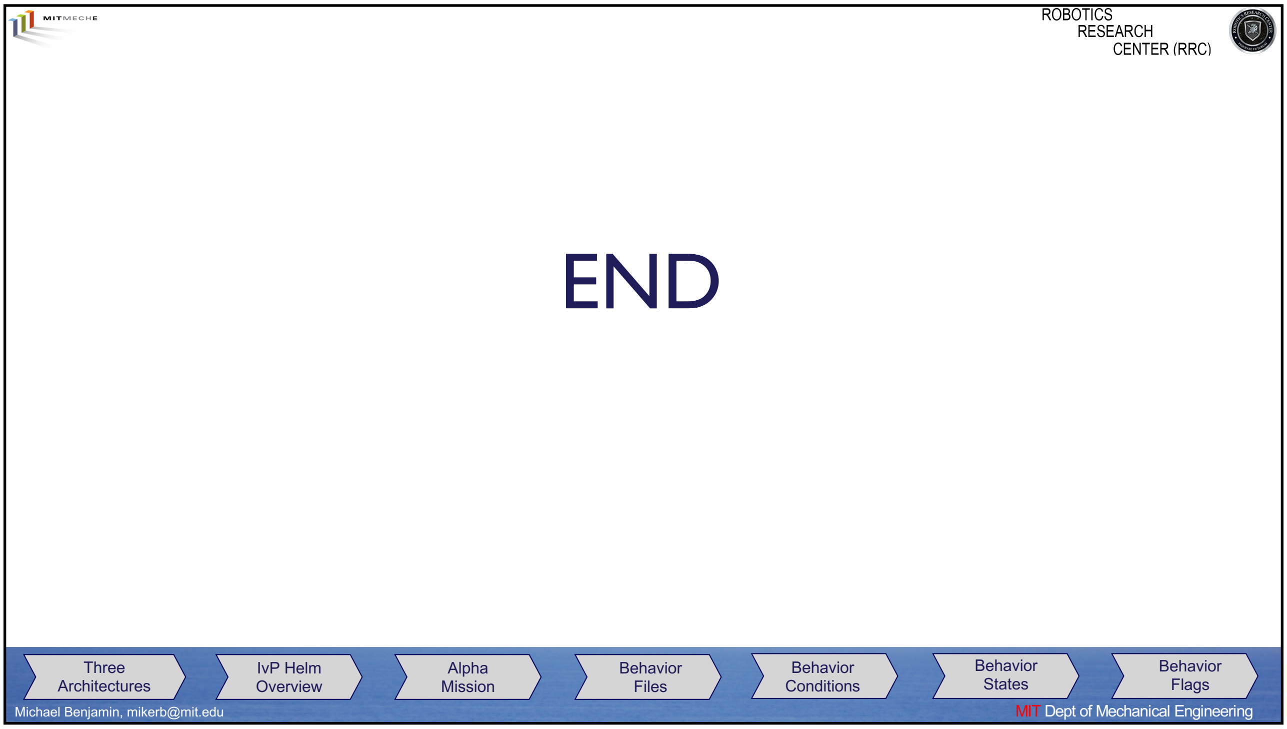
ROBOTICS RESEARCH CENTER (RRC)

Aquaticus Charlie Mission

```
$ cd missions_minicourse/charlie
$ ./launch_sim.sh 8
```

Three Architectures | IvP Helm Overview | Alpha Mission | Behavior Files | Behavior Conditions | Behavior States | Behavior Flags

58



MITMECHE

ROBOTICS
RESEARCH
CENTER (RRC)

END

Three Architectures | IvP Helm Overview | Alpha Mission | Behavior Files | Behavior Conditions | Behavior States | Behavior Flags

Michael Benjamin, mikerb@mit.edu

MIT Dept of Mechanical Engineering

59