

MIT 2.680
UNMANNED MARINE VEHICLE AUTONOMY,
SENSING, AND COMMUNICATIONS

Lecture 8 – Intro to the IvP Helm and Behaviors

March 12th, 2024

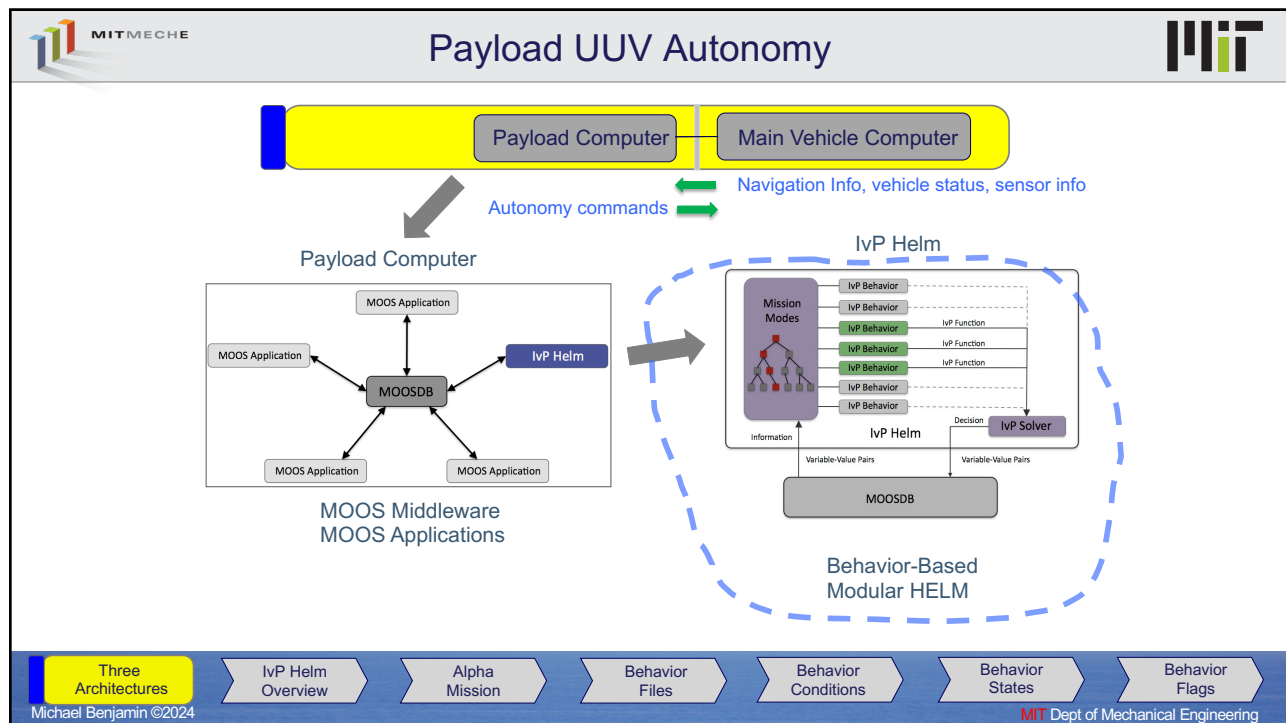
Web: <http://oceanai.mit.edu/2.680>

Email:
Mike Benjamin, mikerb@mit.edu



MIT 2.680 Spring 2024 – Marine Autonomy – "Introduction to the IvP Helm and Behaviors"

Photo by Arjan Vermeij
GLINT '09

1



2

IvP Helm Overview

- Behavior Files
- Behavior Conditions
- Behavior States
- Behavior Flags

Three Architectures

IvP Helm Overview

Alpha Mission

Behavior Files

Behavior Conditions



Behavior States

Behavior Flags

MIT Dept of Mechanical Engineering

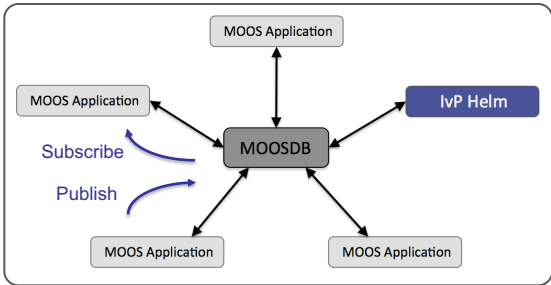
Michael Benjamin ©2024

3

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.



```

graph TD
    subgraph "A MOOS Community"
        direction TB
        A1[MOOS Application] <--> MOOSDB[MOOSDB]
        A2[MOOS Application] <--> MOOSDB
        A3[MOOS Application] <--> MOOSDB
        A4[MOOS Application] <--> MOOSDB
        A5[IvP Helm] <--> MOOSDB
        A1 -- Publish --> MOOSDB
        A2 -- Subscribe --> MOOSDB
    end

```

A MOOS Community

Three Architectures

IvP Helm Overview

Alpha Mission

Behavior Files

Behavior Conditions

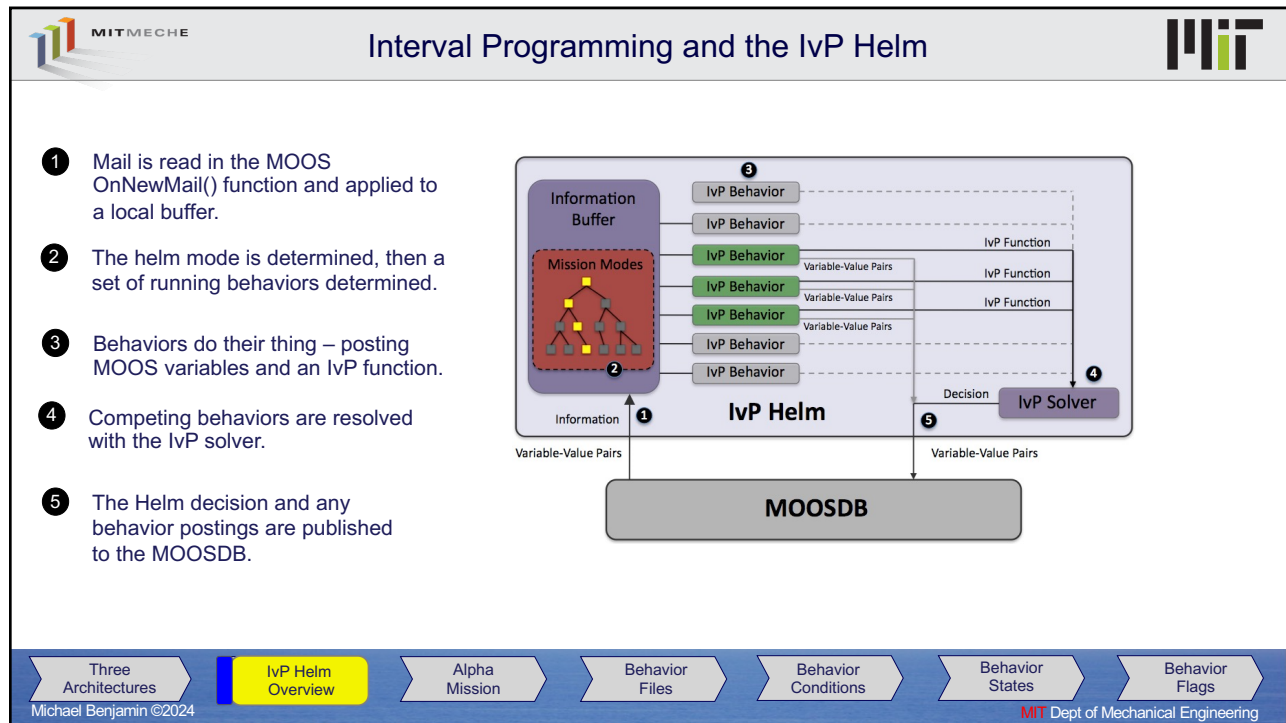
Behavior States

Behavior Flags

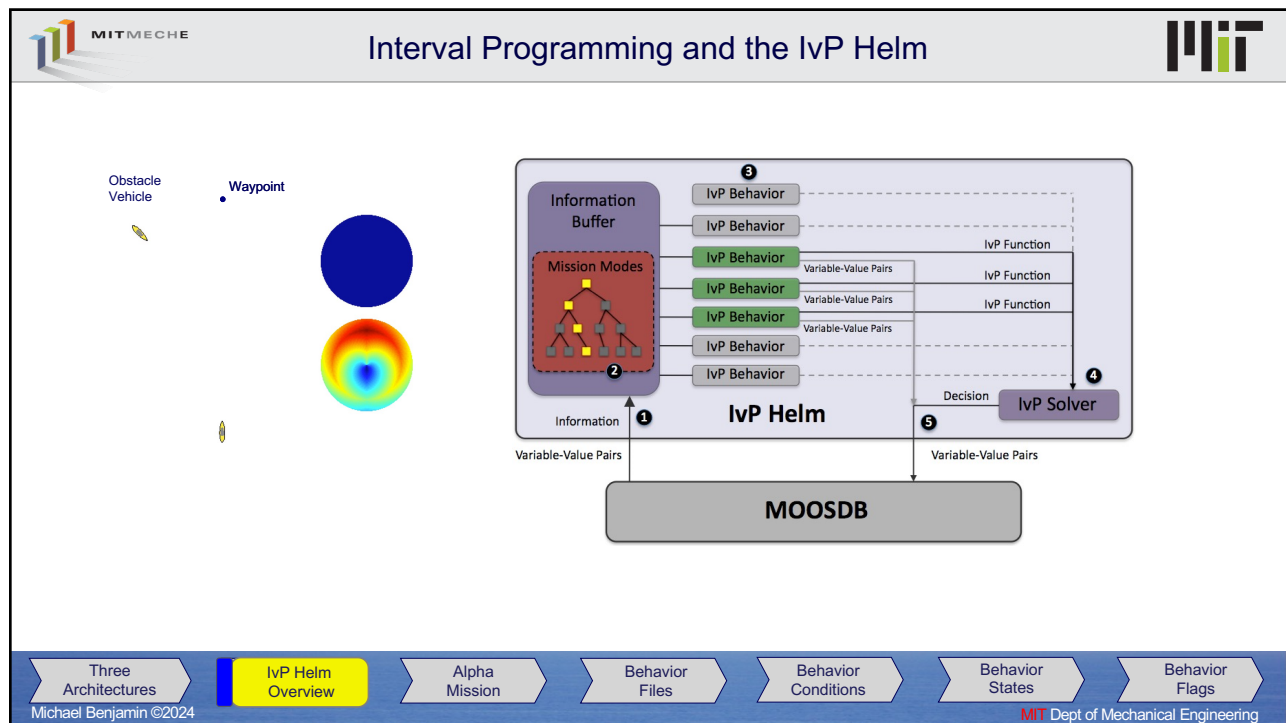
MIT Dept of Mechanical Engineering

Michael Benjamin ©2024

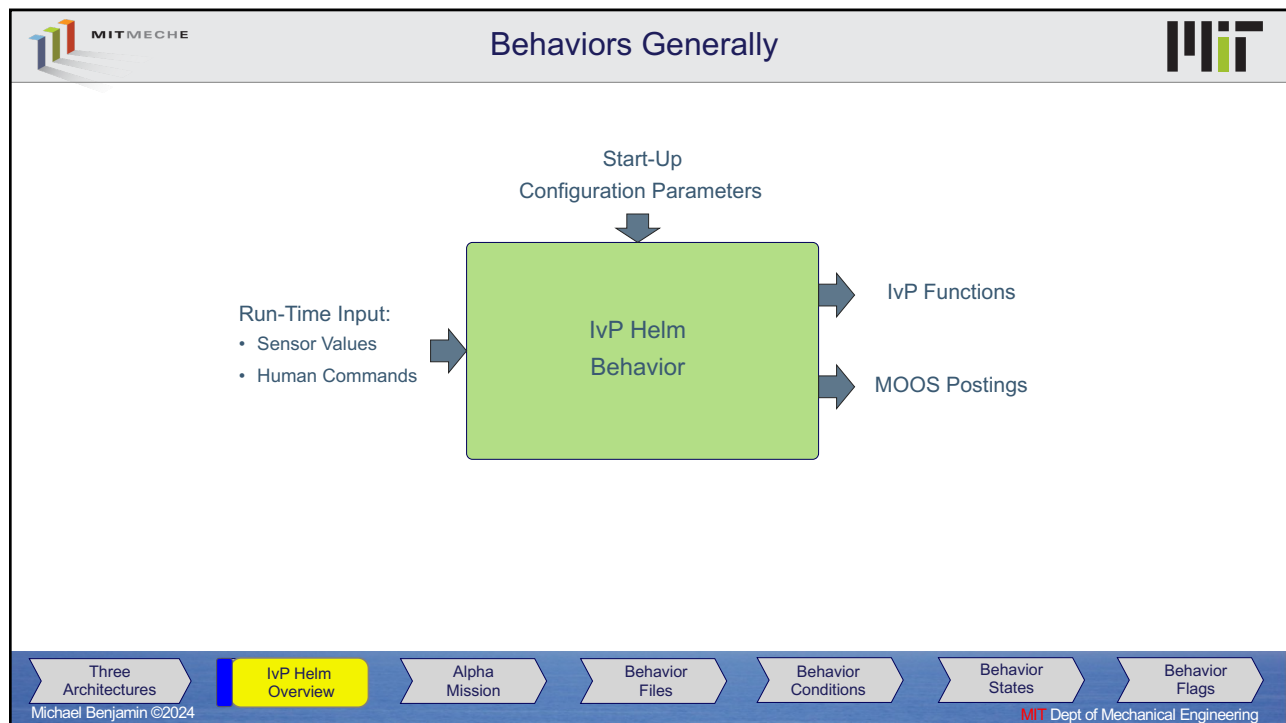
4



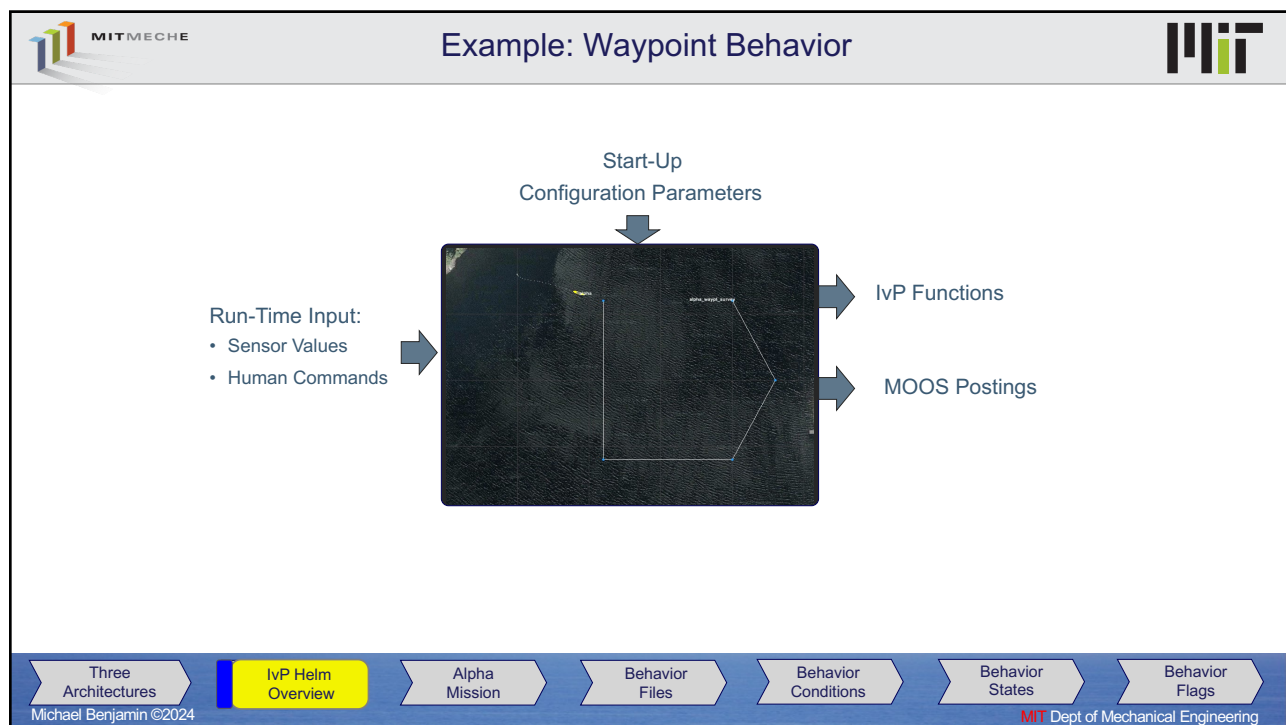
5



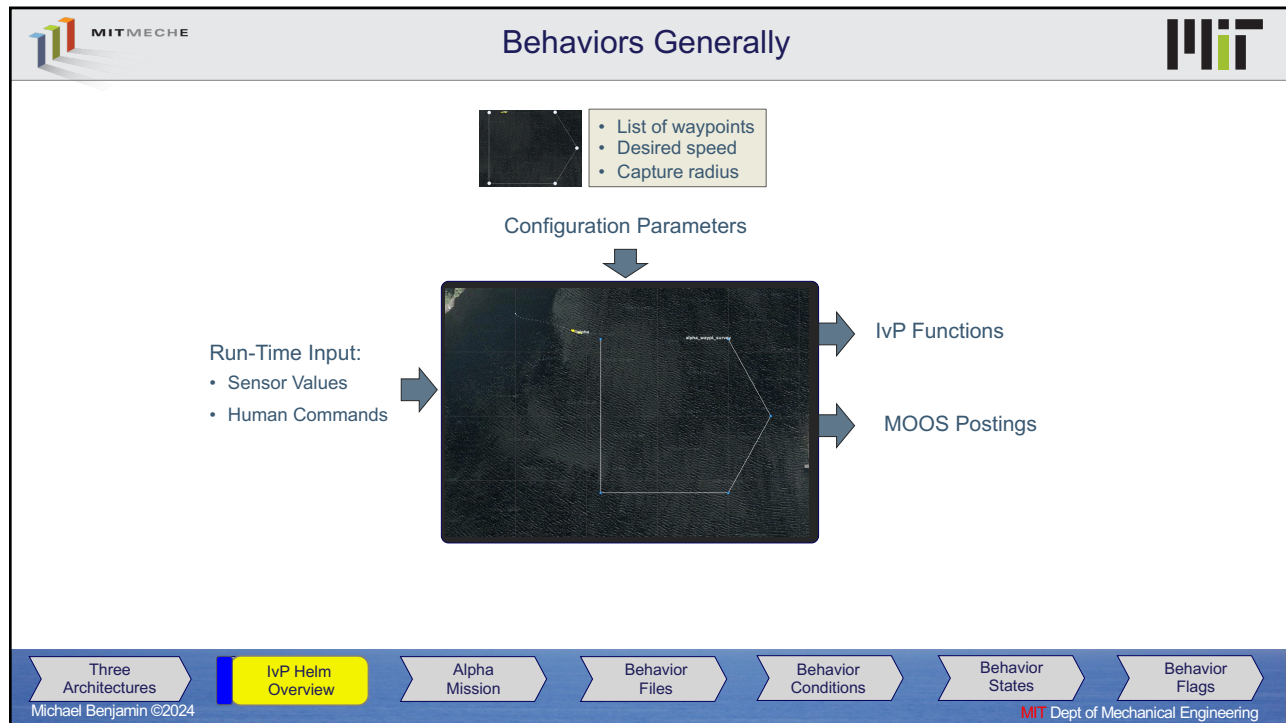
6



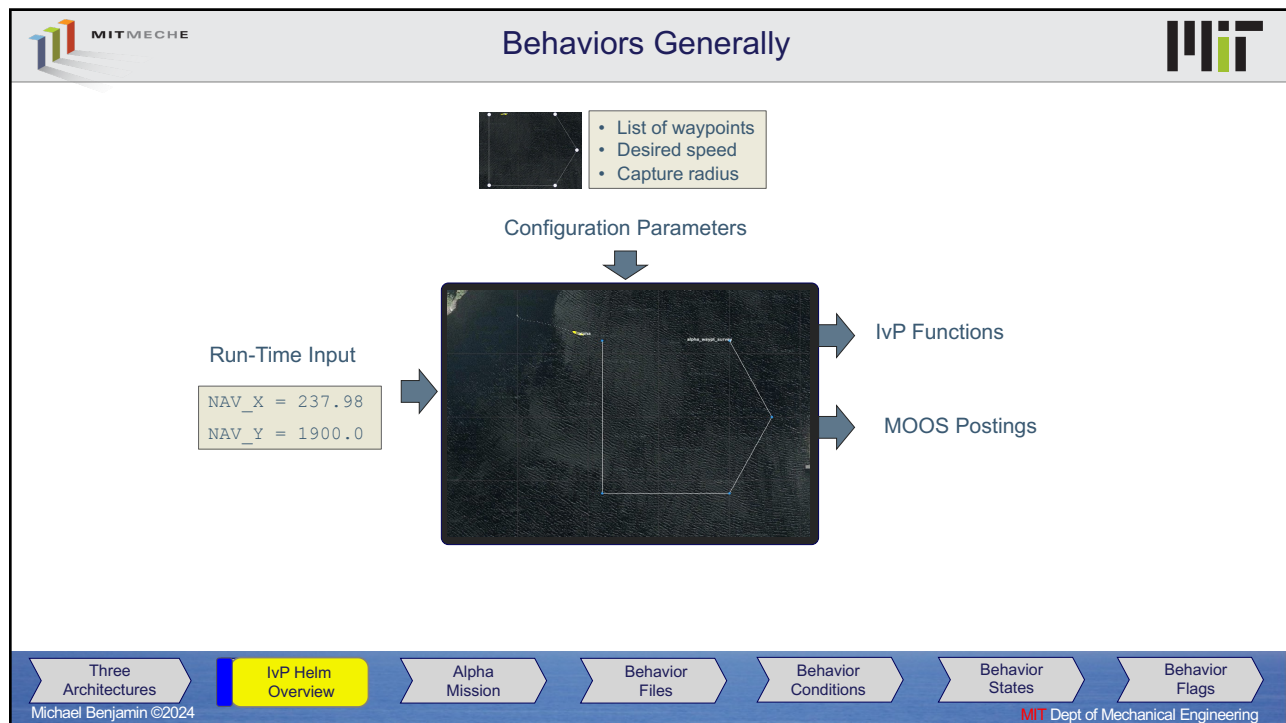
7



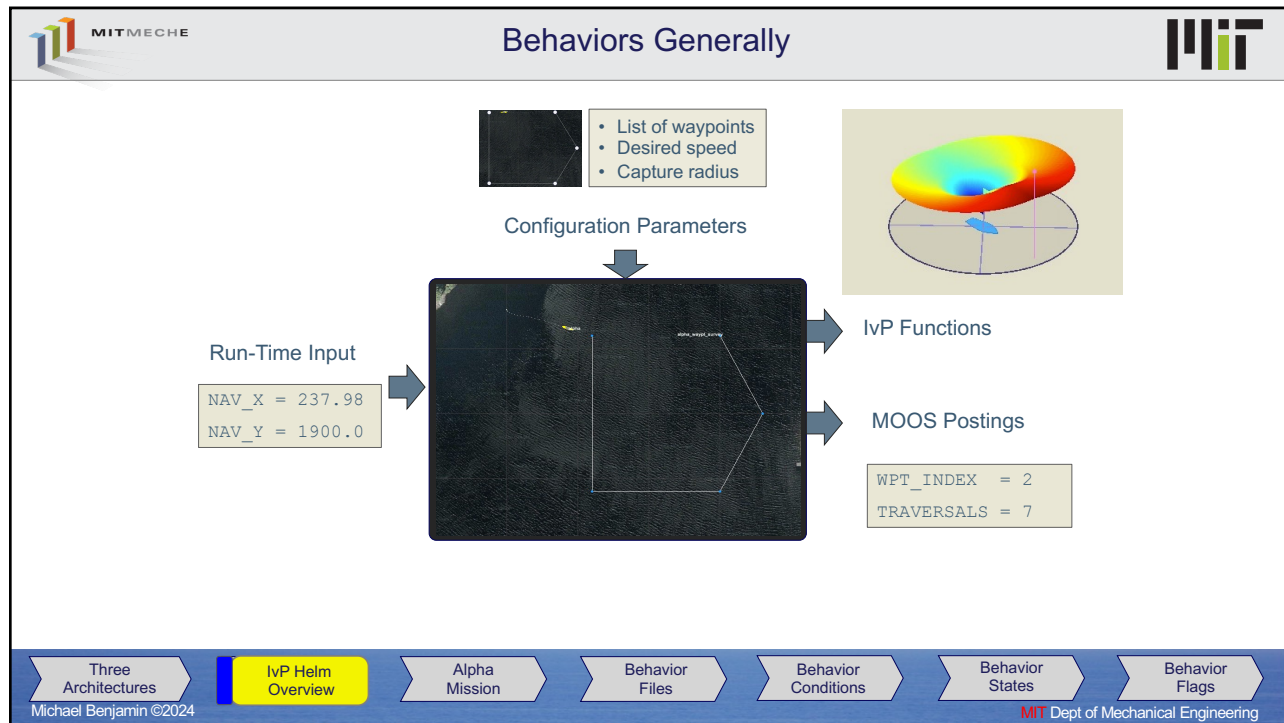
8



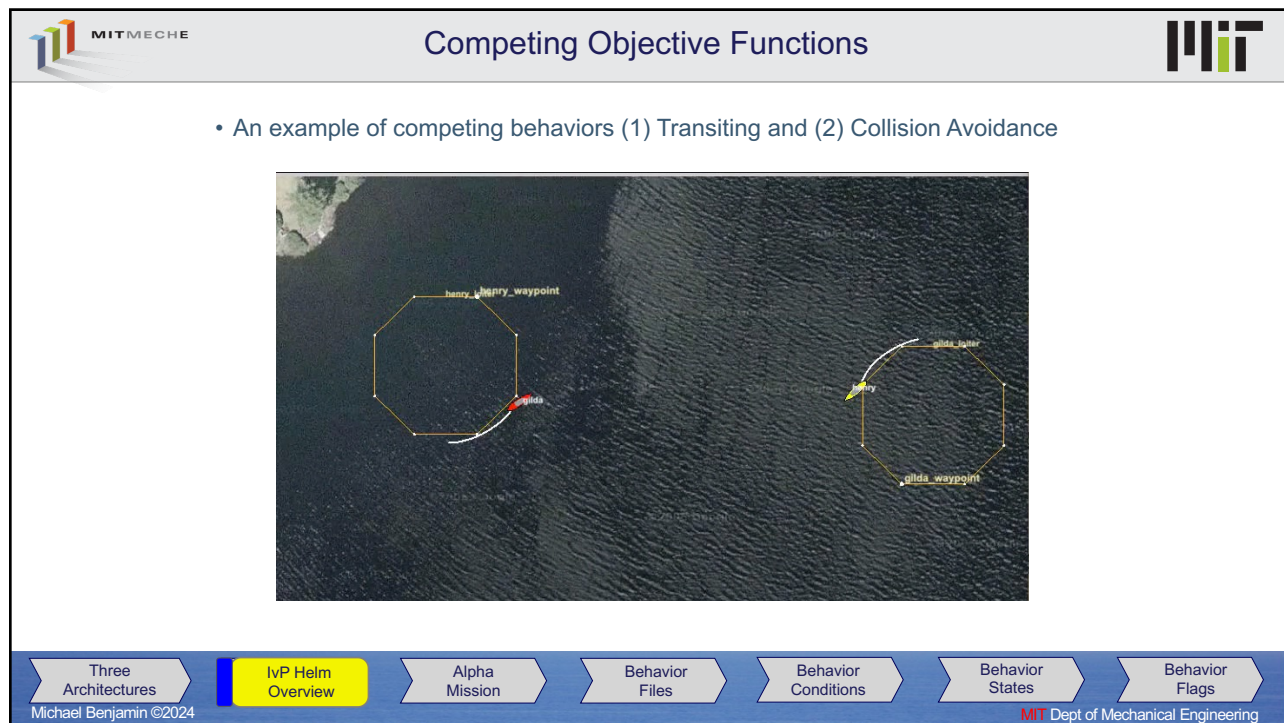
9




10




11




12

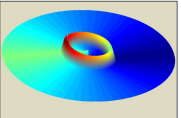


Competing Objective Functions

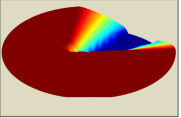


- Each vehicle is running two behaviors
- Each produces its own objective function





Transiting
Objective Function



Collision Avoidance
Objective Function

Three Architectures

IvP Helm Overview

Alpha Mission

Behavior Files


Behavior Conditions

Behavior States


Behavior Flags

MIT Dept of Mechanical Engineering


13

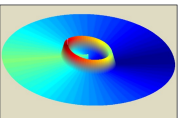
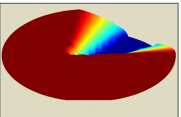


Competing Objective Functions

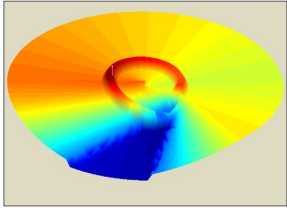


- Each vehicle is running two behaviors
- Each produces its own objective function



Individual
Objective Functions



Collective
Objective Function

Three Architectures

IvP Helm Overview

Alpha Mission

Behavior Files

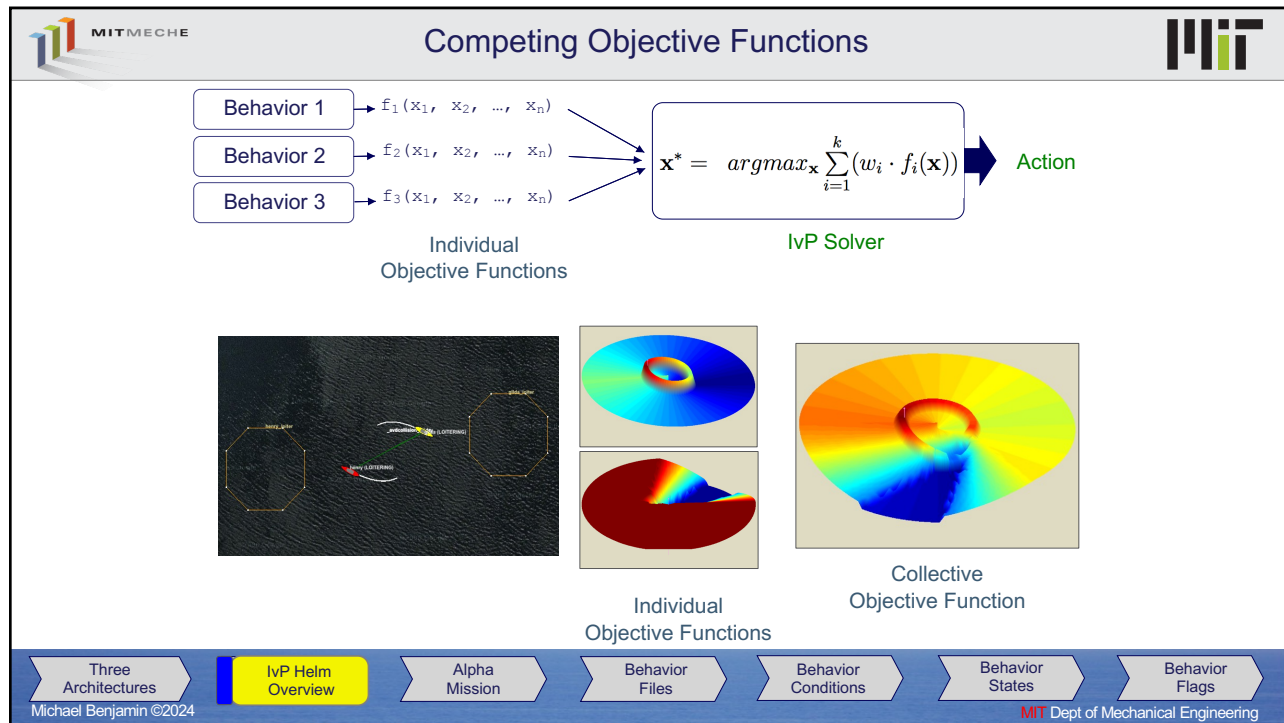
Behavior Conditions

Behavior States

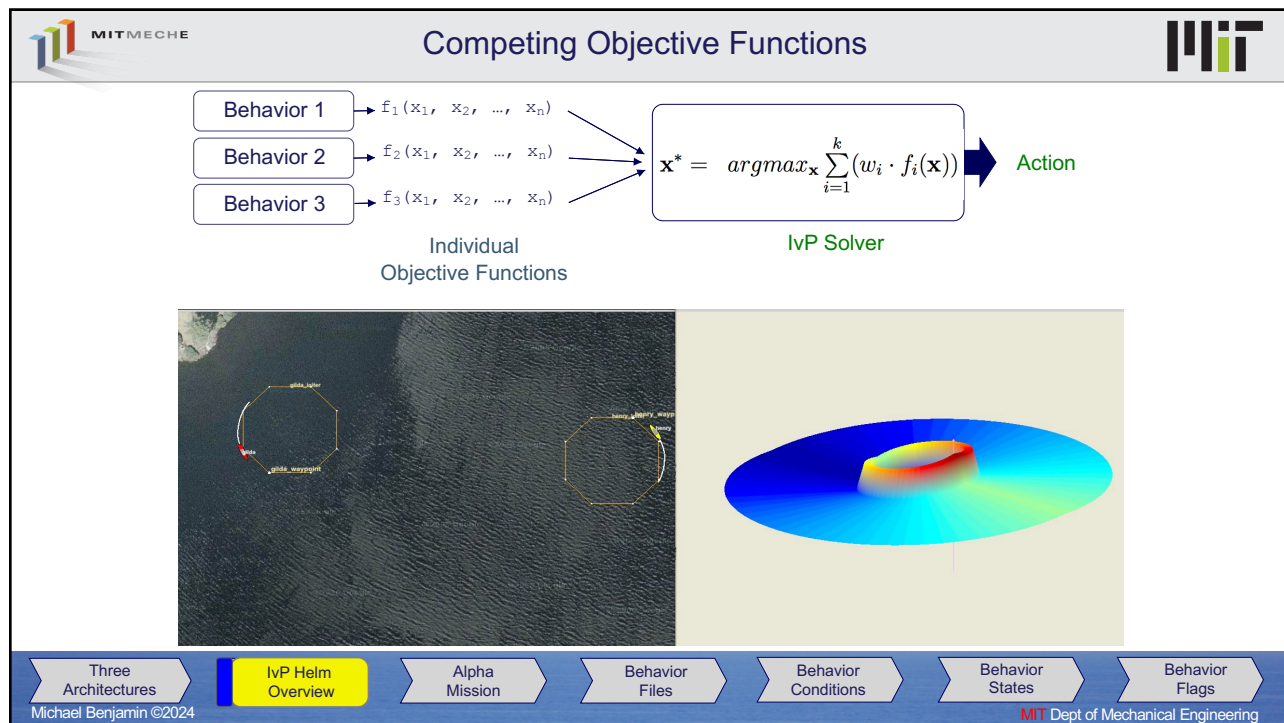
Behavior Flags

MIT Dept of Mechanical Engineering


14




15



16

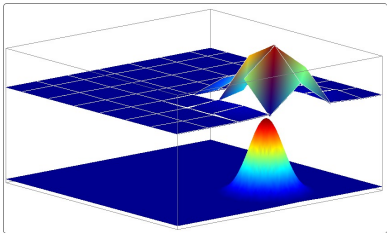


Interval Programming

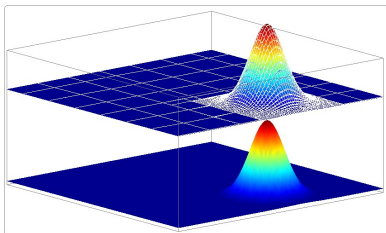


- IvP is Interval Programming
- It is a format for representing objective functions
- It is a solver that capitalizes on that format – fast, globally optimal

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


MIT Dept of Mechanical Engineering

Michael Benjamin ©2024

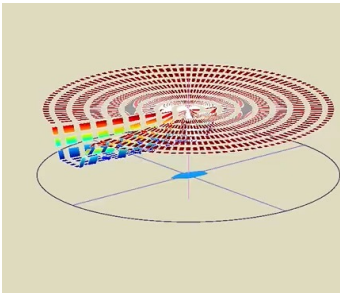

17



Collision Avoidance IvP Functions



- The rendered IvP function below is based on closest point of approach (CPA)
- Note the uniform distribution of pieces.
- Uniform pieces are fine, but computationally inefficient in capturing plateaus.

Three Architectures

IvP Helm Overview

Alpha Mission

Behavior Files

Behavior Conditions


Behavior States

Behavior Flags


MIT Dept of Mechanical Engineering

Michael Benjamin ©2024

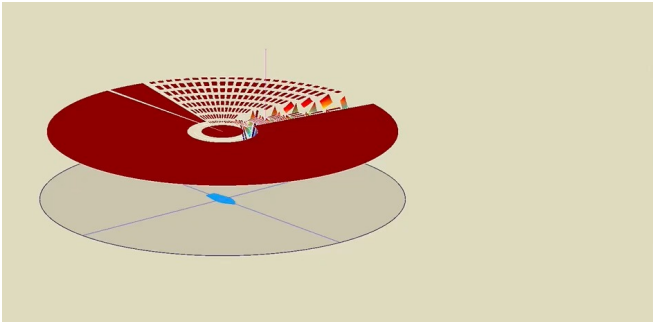

18



Collision Avoidance IvP Functions



- The IvP functions need not be uniform
- The plateau regions can be identified in real time and replace many smaller pieces
- Note the large plateaus in the IvP function below.

Three Architectures

IvP Helm Overview

Alpha Mission

Behavior Files

Behavior Conditions


Behavior States


Behavior Flags

MIT Dept of Mechanical Engineering

Michael Benjamin ©2024

19





The Alpha Mission

Three Architectures

IvP Helm Overview

Alpha Mission

Behavior Files

Behavior Conditions

Behavior States

Behavior Flags

MIT Dept of Mechanical Engineering

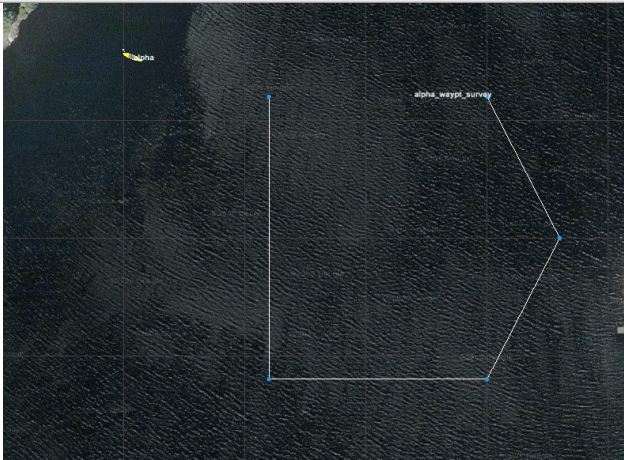
Michael Benjamin ©2024

20

MITMECHE

The Alpha Mission

MIT



To launch yourself:

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

Three Architectures

IvP Helm Overview

Alpha Mission

Behavior Files

Behavior Conditions

Behavior States

Behavior Flags

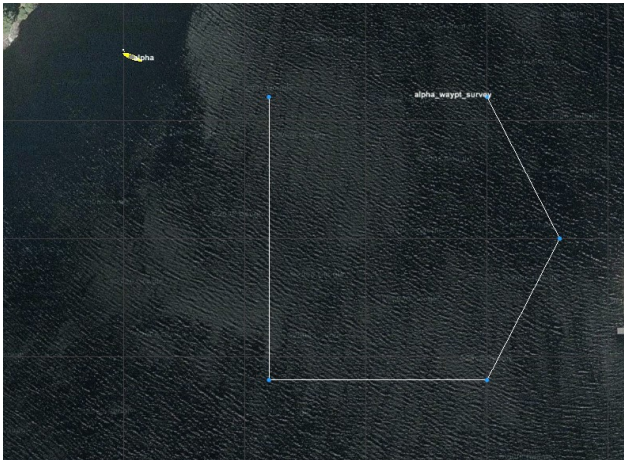
MIT Dept of Mechanical Engineering

21

MITMECHE

Alpha Mission Has Two Behaviors

MIT



```

graph TD
    START([START]) --> Survey[Waypoint Behavior<br/>(surveying)]
    Survey --> Return[Waypoint Behavior<br/>(returning)]
    Return --> STOP{{STOP}}
  
```

Three Architectures

IvP Helm Overview

Alpha Mission

Behavior Files


Behavior Conditions

Behavior States


Behavior Flags

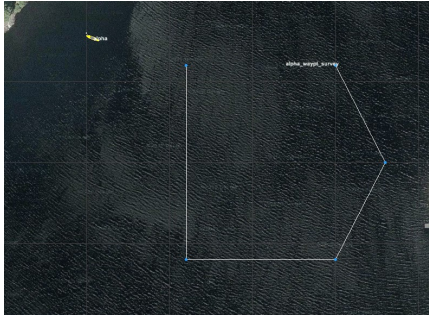
MIT Dept of Mechanical Engineering

22



Alpha Mission Has Two Behaviors





```

graph TD
    START([START]) --> WB[Waypoint Behavior<br/>(surveying)]
    WB --> WR[Waypoint Behavior<br/>(returning)]
    WR --> STOP([STOP])
  
```

Three questions discussed next:

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

Three Architectures

IvP Helm Overview

Alpha Mission

Behavior Files


Behavior Conditions

Behavior States


Behavior Flags

MIT Dept of Mechanical Engineering

23

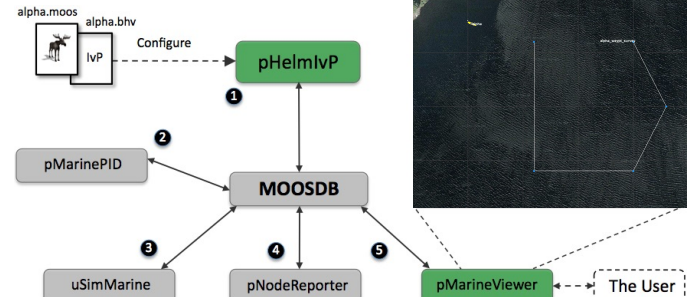



Configuring the Helm for a Mission



A mission is configured with two files:

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

To launch it yourself:

```

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

Three Architectures

IvP Helm Overview

Alpha Mission

Behavior Files

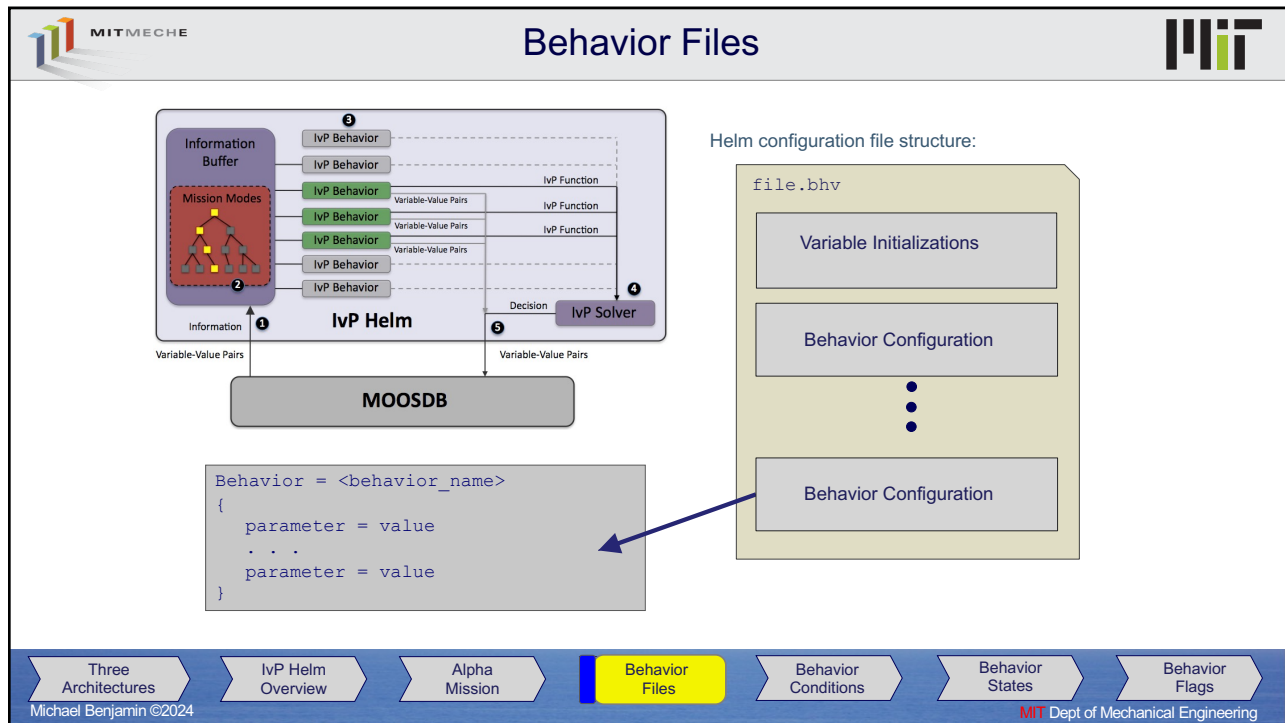
Behavior Conditions

Behavior States

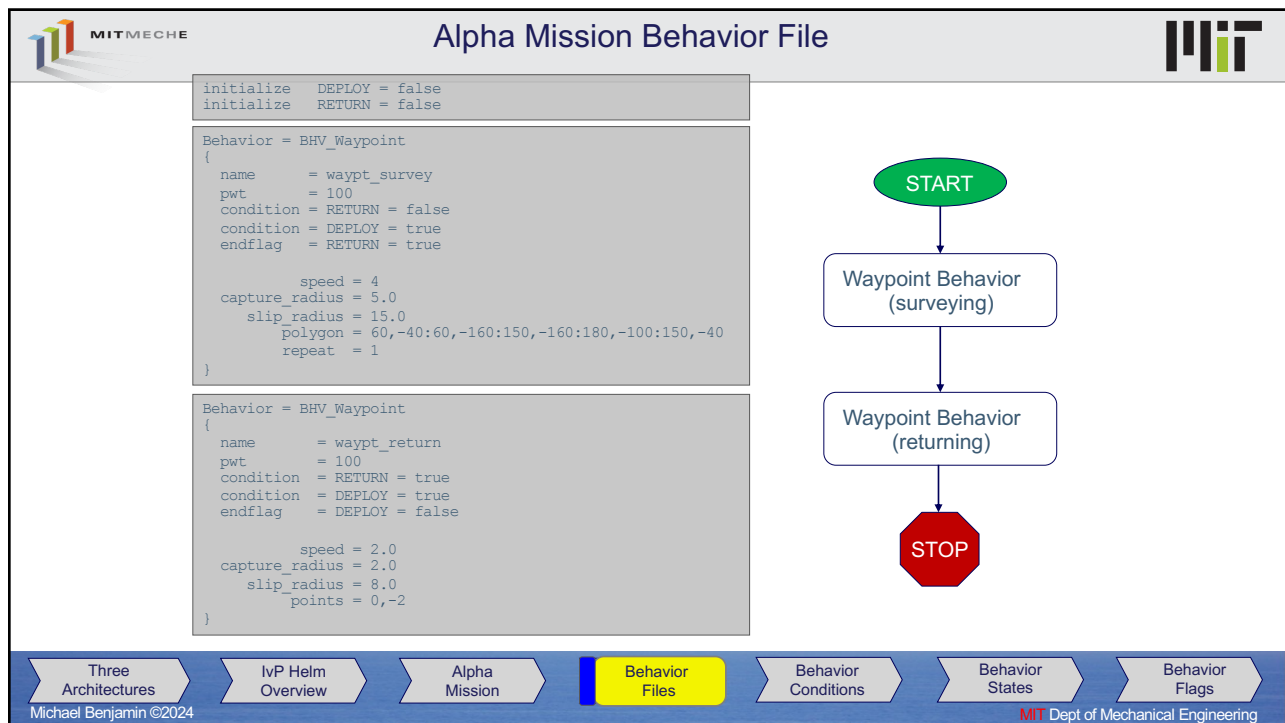
Behavior Flags

MIT Dept of Mechanical Engineering


24




25



26



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

```

Three Architectures

IvP Helm Overview

Alpha Mission

Behavior Files

Behavior Conditions


Behavior States

Behavior Flags


MIT Dept of Mechanical Engineering

Michael Benjamin ©2024

27



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


MIT Dept of Mechanical Engineering

Michael Benjamin ©2024

28



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 ©2024
MIT Dept of Mechanical Engineering

29




Behavior Conditions




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

Michael Benjamin ©2024
MIT Dept of Mechanical Engineering

30



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 and DEPLOY are MOOS variables
- Both are of type string (not double)(No such thing as Boolean in MOOS variable types)
- The condition is true if the current variable value matches the string

Three Architectures

IvP Helm Overview

Alpha Mission

Behavior Files


Behavior Conditions

Behavior States


Behavior Flags

MIT Dept of Mechanical Engineering

31



Behavior Conditions



- Each condition involves one or more MOOS variables
- A behavior may have more than one condition
- If there are 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
}

```

Three Architectures

IvP Helm Overview

Alpha Mission

Behavior Files


Behavior Conditions

Behavior States


Behavior Flags

MIT Dept of Mechanical Engineering

32



Behavior Conditions



- Each condition involves one or more MOOS variables
- A behavior may have more than one condition
- If there are 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
}

```

Same as:

condition = (RETURN=false) and (DEPLOY=true)

Three Architectures

IvP Helm Overview

Alpha Mission

Behavior Files


Behavior Conditions

Behavior States


Behavior Flags

MIT Dept of Mechanical Engineering

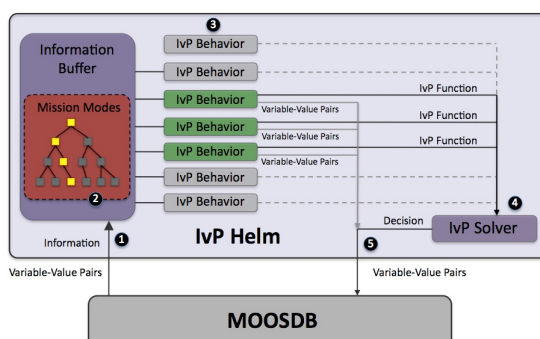
33



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



Three Architectures

IvP Helm Overview

Alpha Mission

Behavior Files

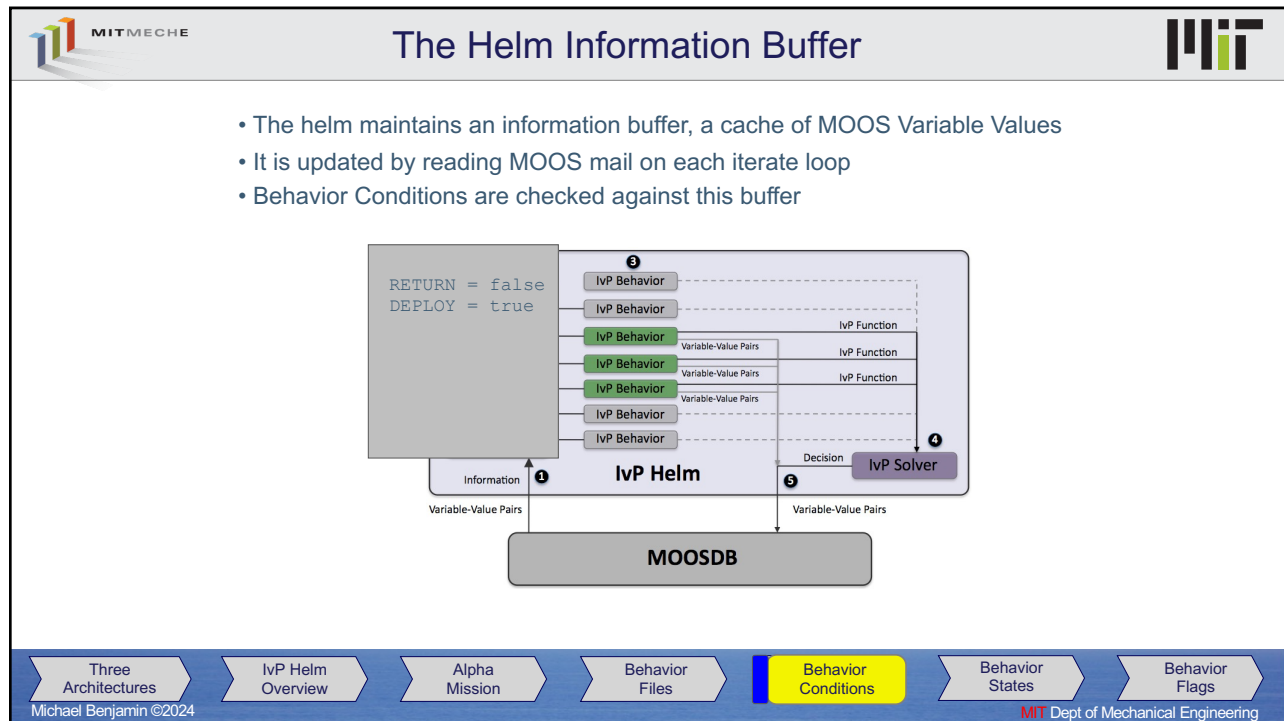
Behavior Conditions

Behavior States

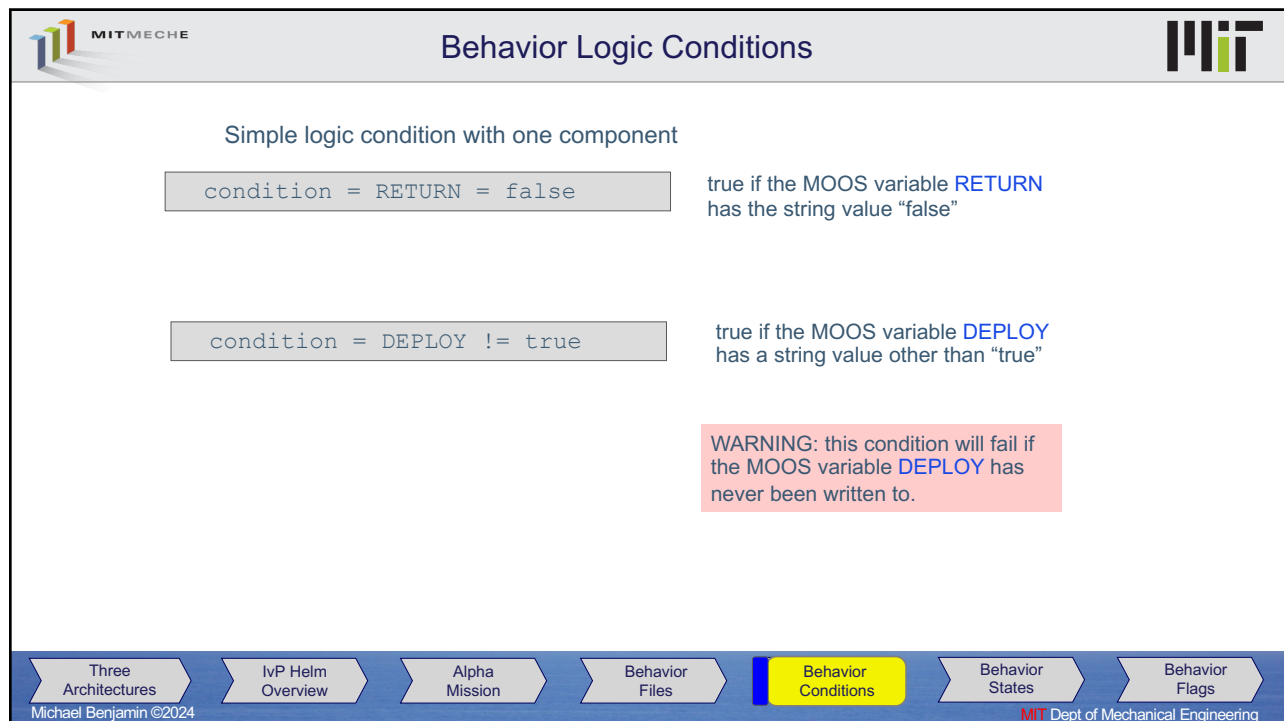
Behavior Flags

MIT Dept of Mechanical Engineering


34




35



36



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

Three Architectures

IvP Helm Overview

Alpha Mission

Behavior Files


Behavior Conditions

Behavior States


Behavior Flags

MIT Dept of Mechanical Engineering

37



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
}
```

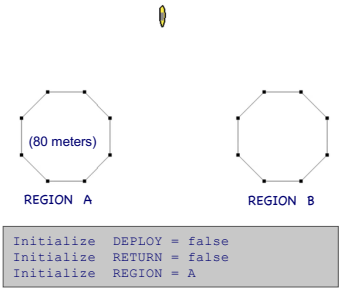
```
Behavior = BHV_Loiter
{
  name      = loiter_b
  condition = ((DEPLOY=true)and(REGION=B)) and (RETURN=false)

  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



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

Three Architectures

IvP Helm Overview

Alpha Mission

Behavior Files


Behavior Conditions

Behavior States


Behavior Flags

MIT Dept of Mechanical Engineering

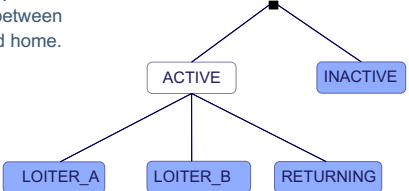
38



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.



```

graph TD
    Root(( )) --> ACTIVE[ACTIVE]
    Root --> INACTIVE[INACTIVE]
    ACTIVE --> LOITER_A[LOITER_A]
    ACTIVE --> LOITER_B[LOITER_B]
    ACTIVE --> RETURNING[RETURNING]
          
```

Launch and return position



REGION A REGION B

Three Architectures

IvP Helm Overview

Alpha Mission

Behavior Files

Behavior Conditions


Behavior States

Behavior Flags


MIT Dept of Mechanical Engineering

Michael Benjamin ©2024

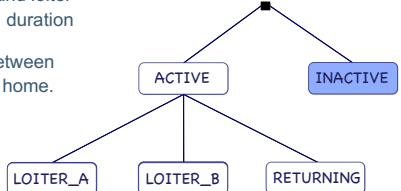
39



Simple Example: "Double Loiter"



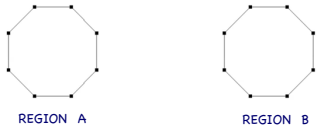
Mission Synopsis:
 Upon receiving a deploy command, transit to and loiter at region A for a fixed duration and then to region B. Periodically switch between regions until recalled home.



```

graph TD
    Root(( )) --> ACTIVE[ACTIVE]
    Root --> INACTIVE[INACTIVE]
    ACTIVE --> LOITER_A[LOITER_A]
    ACTIVE --> LOITER_B[LOITER_B]
    ACTIVE --> RETURNING[RETURNING]
          
```

Launch and return position



REGION A REGION B

animation

```

set MODE = ACTIVE {
  DEPLOY = true
} INACTIVE

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

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

file.bhv

Variable Initializations

Hierarchical Mode Declarations

Behavior Configurations

Three Architectures

IvP Helm Overview

Alpha Mission

Behavior Files

Behavior Conditions

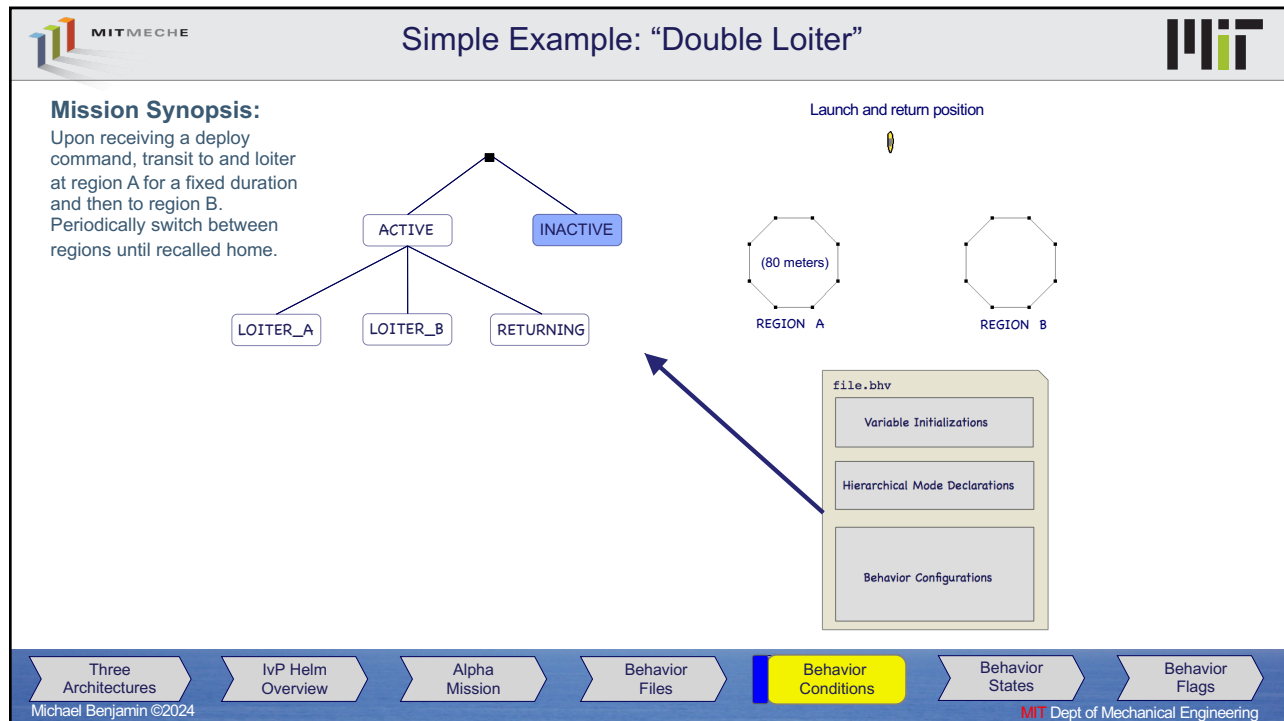
Behavior States

Behavior Flags

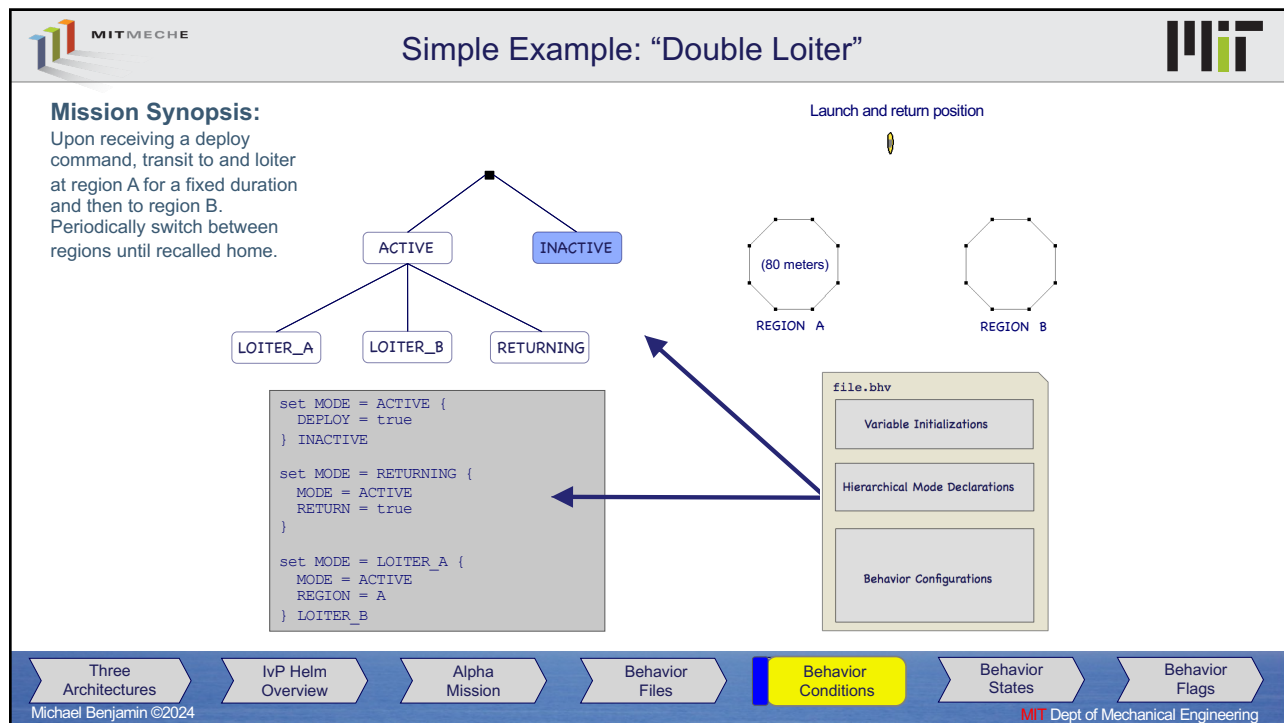
MIT Dept of Mechanical Engineering

Michael Benjamin ©2024


40




41



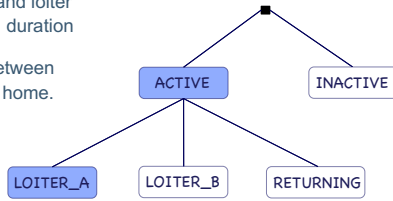
42



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.



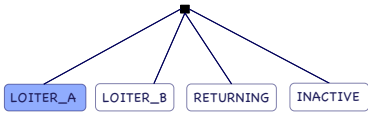
```

set MODE = ACTIVE {
  DEPLOY = true
} INACTIVE

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

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

Question:
Why define the "Active" mode?
Why not just have:



```

file.bhv
Variable Initializations
Hierarchical Mode Declarations
Behavior Configurations
```

←

Three Architectures

IvP Helm Overview

Alpha Mission

Behavior Files


Behavior Conditions

Behavior States


Behavior Flags

MIT Dept of Mechanical Engineering

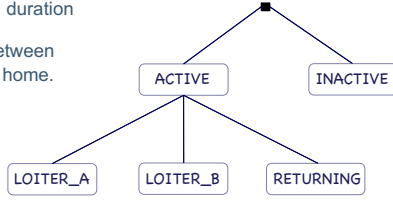
43



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.



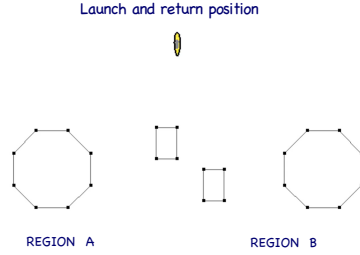
```

set MODE = ACTIVE {
  DEPLOY = true
} INACTIVE

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

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

Launch and return position



```

file.bhv
Variable Initializations
Hierarchical Mode Declarations
Behavior Configurations
```

←

Three Architectures

IvP Helm Overview

Alpha Mission

Behavior Files

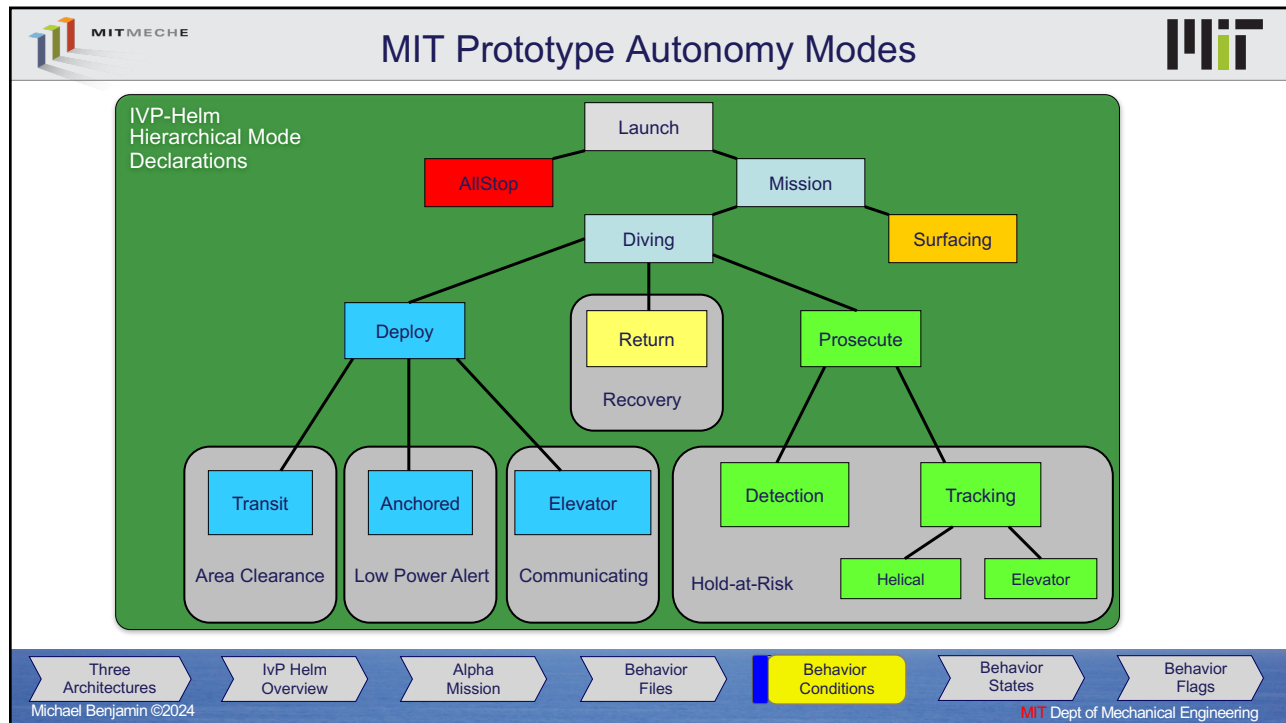
Behavior Conditions

Behavior States

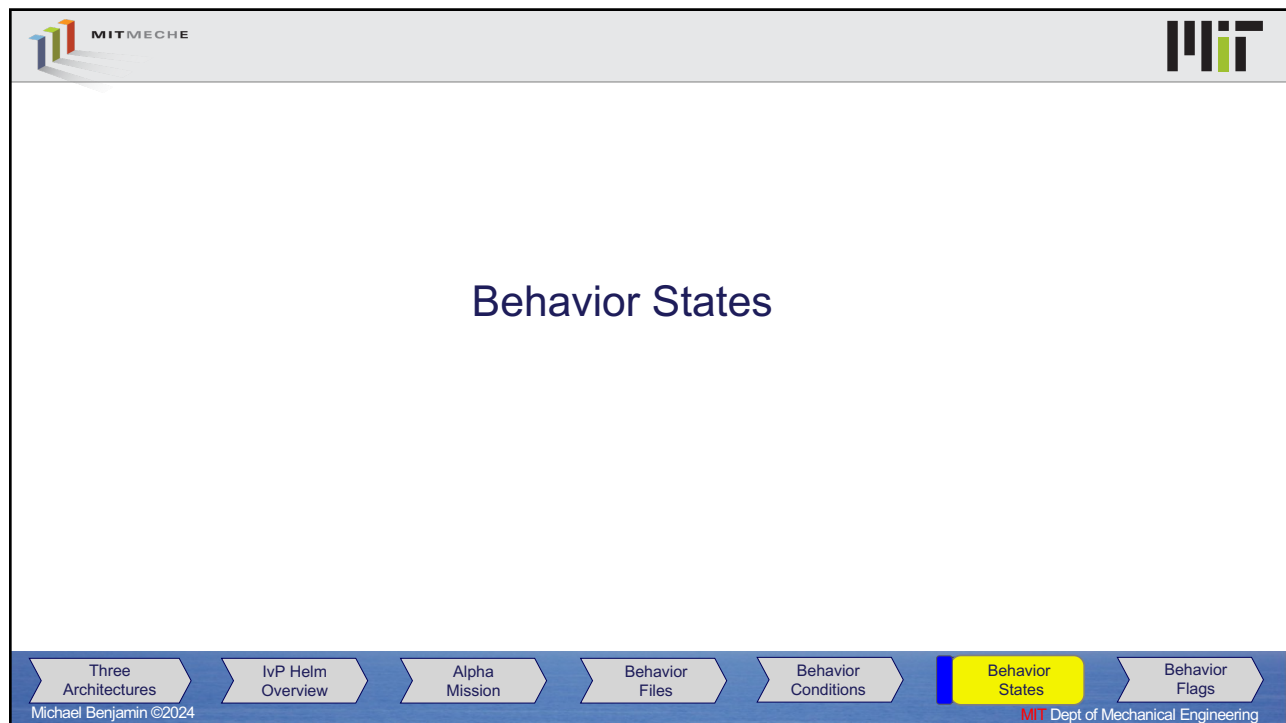
Behavior Flags

MIT Dept of Mechanical Engineering


44




45



46



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 ©2024 MIT Dept of Mechanical Engineering

47



Active vs. Running States

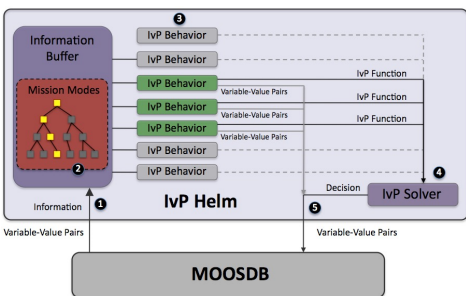


Idle

Running

Active

Completed



The diagram illustrates the IvP Helm architecture. It shows a central 'IvP Helm' block containing 'Mission Modes' and 'IvP Behavior' components. 'Mission Modes' is connected to 'IvP Behavior' via 'Information' and 'Variable-Value Pairs'. 'IvP Behavior' is connected to 'IvP Function' via 'Variable-Value Pairs'. 'IvP Function' is connected to 'IvP Solver' via 'Decision'. 'IvP Solver' is connected to 'MOOSDB' via 'Variable-Value Pairs'. The 'MOOSDB' block is at the bottom, and the 'IvP Helm' block is at the top.

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 ©2024 MIT Dept of Mechanical Engineering

48



Behavior Completion



Behaviors states:

Idle

Running

Active

Completed

Completion is defined by the behavior. For example:

- A **waypoint** behavior *completes* when it has visited all its waypoints.
- A **loiter** behavior never *completes*.

Even behaviors that don't normally *complete*, may complete when configured with a prescribed **duration**, e.g., **duration=60 // seconds**

By default, a completed behavior simply ceases to exist once it is completed. No chance for participation ever again in the helm.

Unless... the behavior is configured with **perpetual=true**.

Three Architectures

IvP Helm Overview

Alpha Mission

Behavior Files

Behavior Conditions

Behavior States

Behavior Flags


MIT Dept of Mechanical Engineering

Michael Benjamin ©2024

49



Behavior Flags



Three Architectures

IvP Helm Overview

Alpha Mission

Behavior Files

Behavior Conditions


Behavior States

Behavior Flags


MIT Dept of Mechanical Engineering

Michael Benjamin ©2024

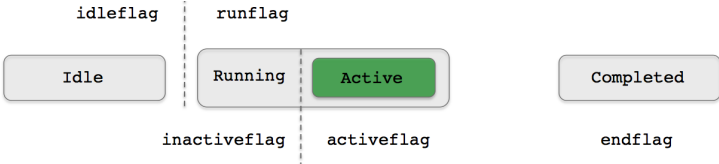
50



Behavior Flags



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



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.

Three Architectures

IvP Helm Overview

Alpha Mission

Behavior Files


Behavior Conditions

Behavior States


Behavior Flags

MIT Dept of Mechanical Engineering

51



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
}

```

Three Architectures

IvP Helm Overview

Alpha Mission

Behavior Files


Behavior Conditions

Behavior States


Behavior Flags

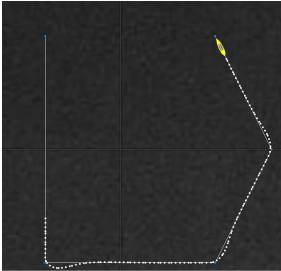
MIT Dept of Mechanical Engineering

52

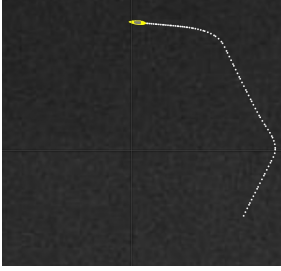


Alpha Mission End Flag Example

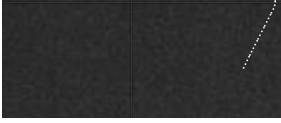




1 survey waypoints completes



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

MIT Dept of Mechanical Engineering

53



Behavior Flags and the Helm Duplication Filter



A **duplication filter** is used by the helm to prevent redundant information from bloating the log files and perhaps overworking consumers of the information.

pHelmIvP

LOITER_REGION = west
 LOITER_REGION = west
 LOITER_REGION = west
 LOITER_REGION = west
 LOITER_REGION = west
 LOITER_REGION = west

MOOSDB

➔

pHelmIvP

duplication filter

LOITER_REGION = west

MOOSDB

- Sometimes we *do* want duplicates.
- Behavior authors have the option of using the filter or not:

postMessage("MSG", "hello");

postRepeatableMessage("MSG", "hello");
- The duplication filter is on by default for all flag postings, e.g., runflag, idleflag, etc.
- The exception is the endflag – duplicates are always posted.**

Three Architectures

IvP Helm Overview

Alpha Mission

Behavior Files

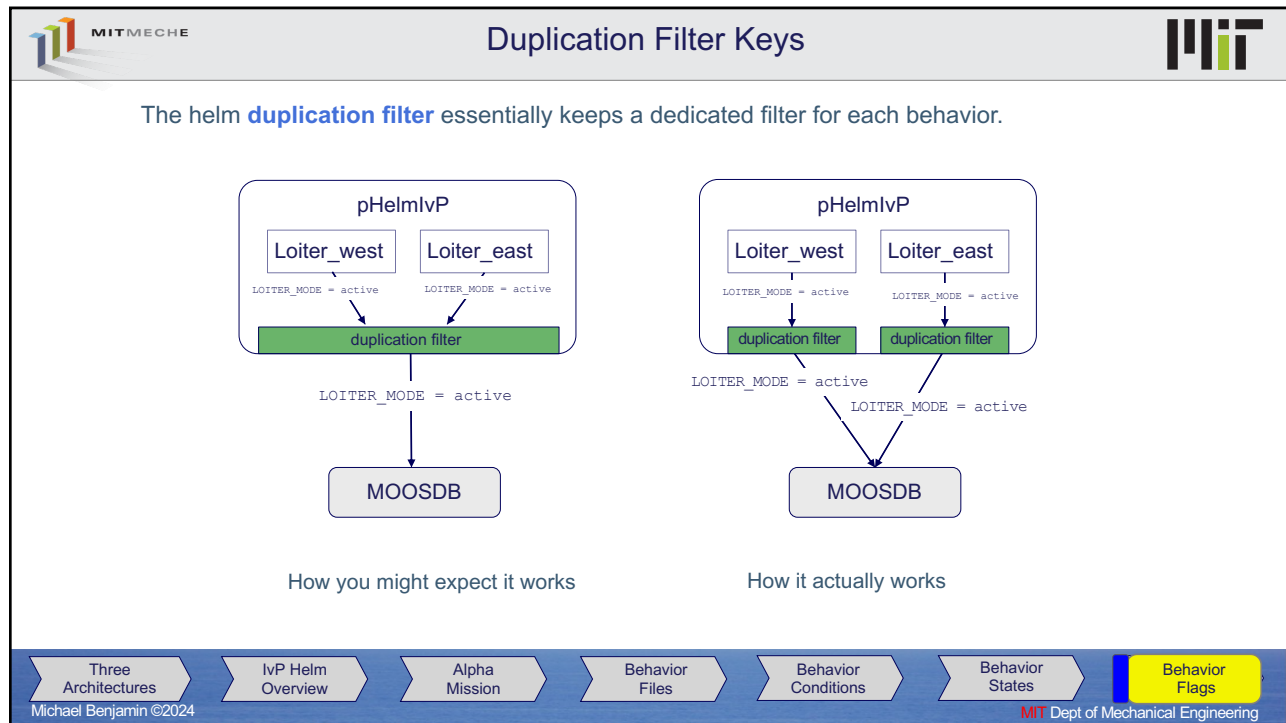
Behavior Conditions

Behavior States

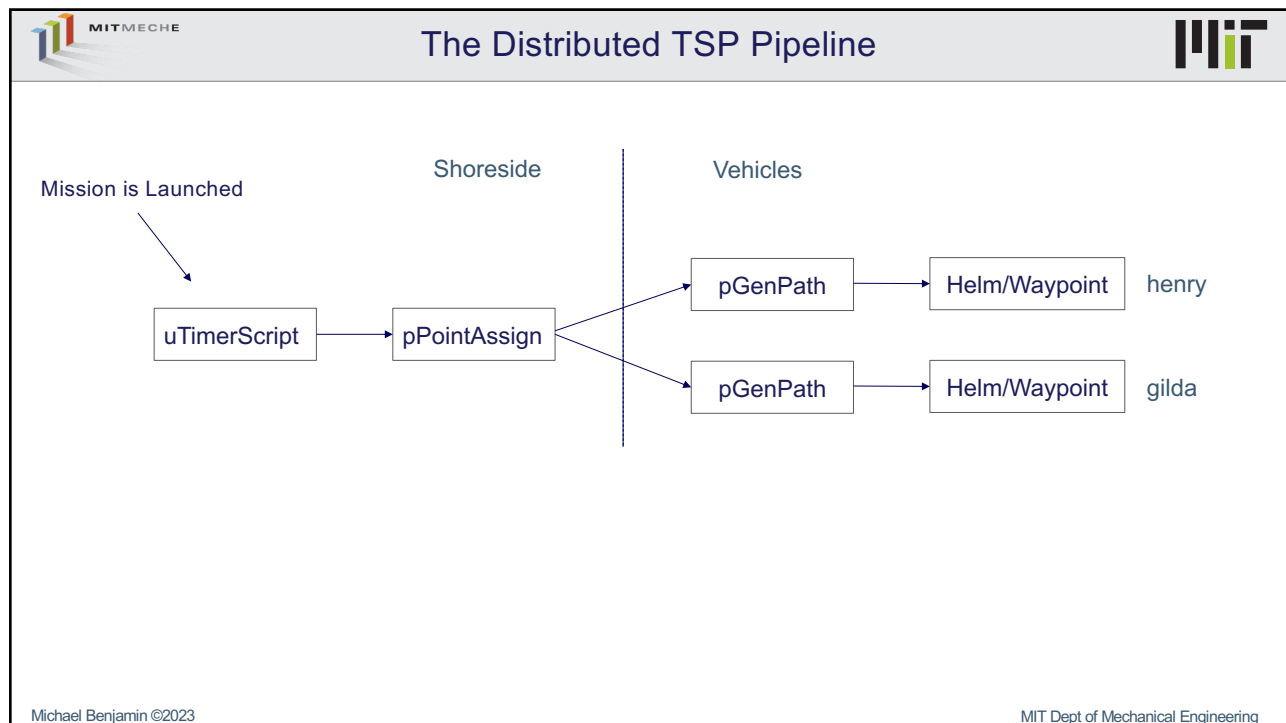
Behavior Flags

MIT Dept of Mechanical Engineering

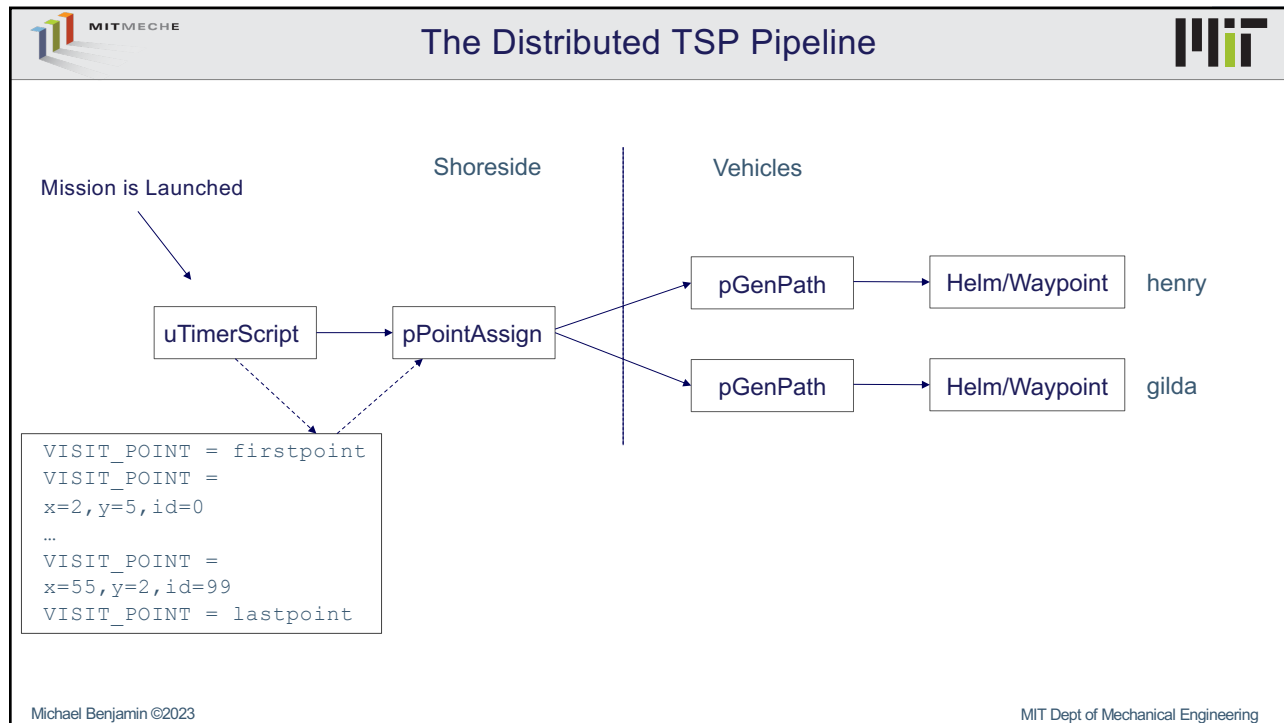
54



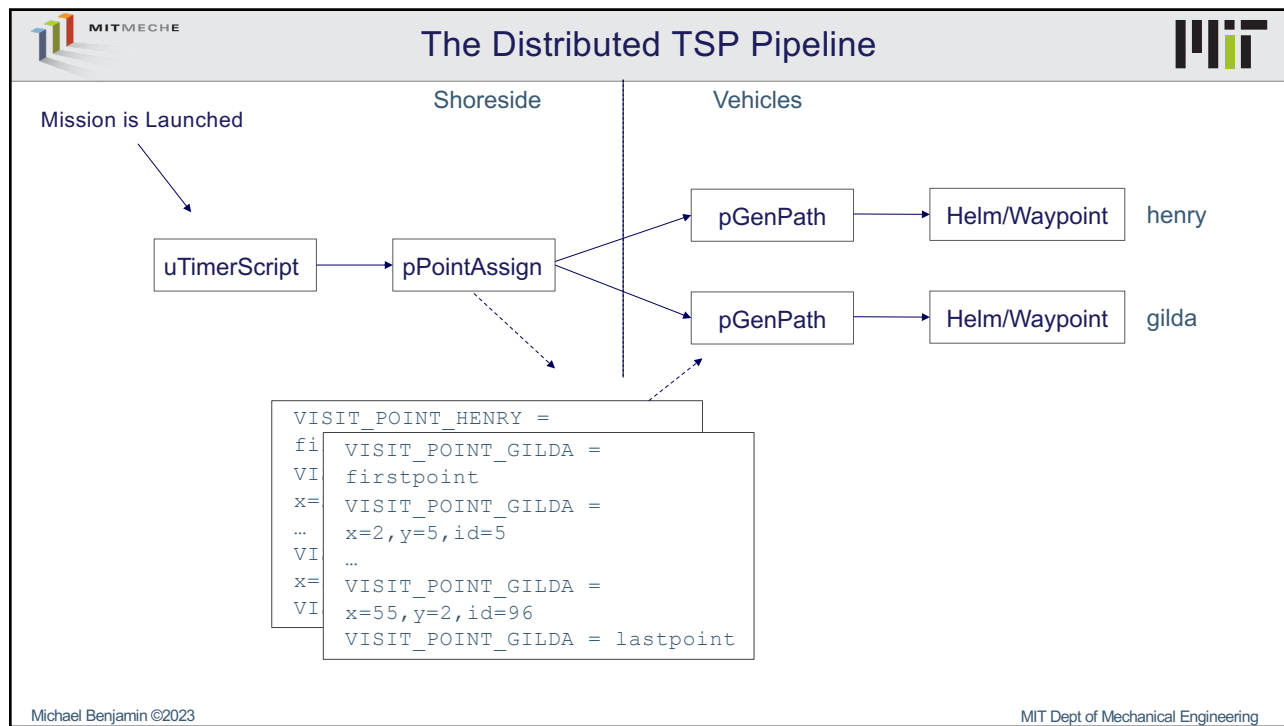
55



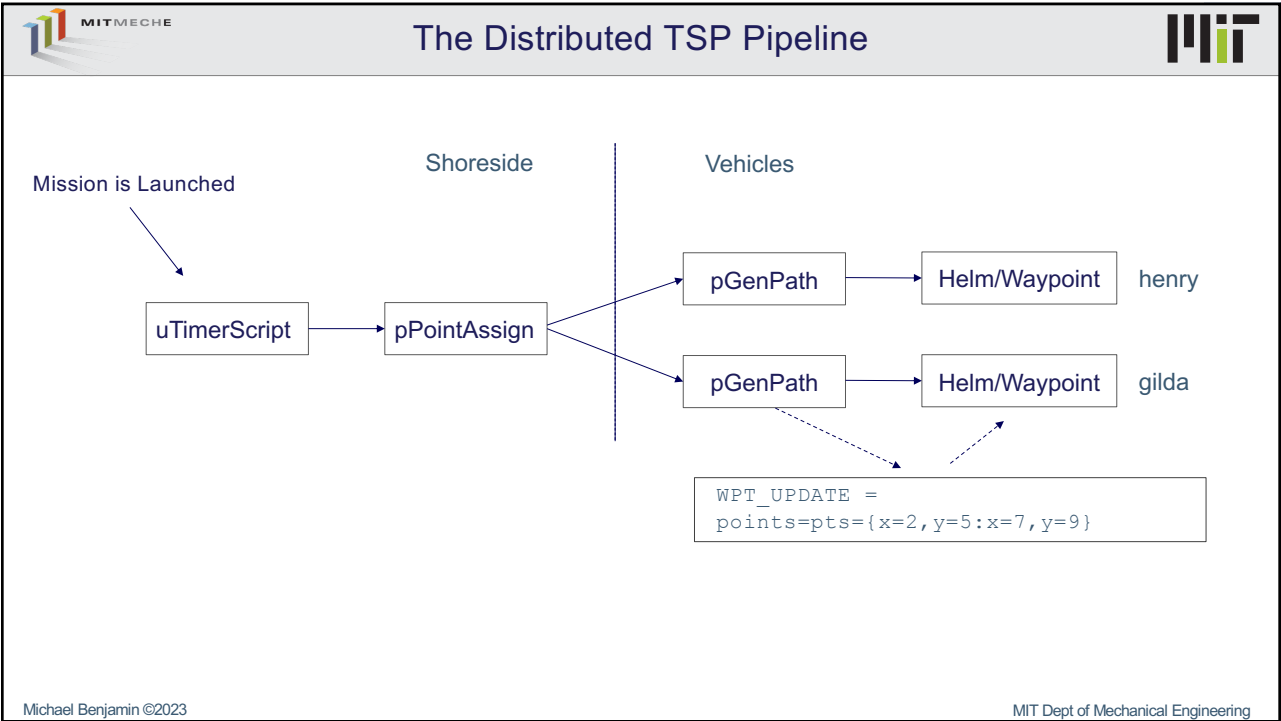
56




57




58





END



Three Architectures

IvP Helm Overview

Alpha Mission

Behavior Files

Behavior Conditions

Behavior States

Behavior Flags

Michael Benjamin ©2024

MIT Dept of Mechanical Engineering