Sampling-Based Motion Planning and Co-Safe LTL for Coverage Missions Using the MOOS-IvP Framework

James McMahon[1,2] and Erion Plaku[2]

[1] U.S. Naval Research Laboratory, Code 7130
Washington, DC

[2] Dept. of Electrical Engineering and Computer Science
Catholic University of America

[This work is funded by the Office of Naval Research (ONR)]

Outline

What we would like to do

mission description in a natural, structured, language

What we would like to do

mission description in a natural, structured, language

- AUV should always be safe
- Objective is to inspect areas $A_1, A_2, \ldots, A_n$
- AUV must adapt to the environment, avoiding any obstacles it encounters
- If a mine is detected, AUV should explore surrounding areas for additional mines
- If a ship is detected, AUV should track it to gather information
- When the mission is completed, AUV should return to the base

What we would like to do

mission description in a natural, structured, language

- AUV should always be safe
- Objective is to inspect areas $A_1, A_2, \ldots, A_n$
- AUV must adapt to the environment, avoiding any obstacles it encounters
- If a mine is detected, AUV should explore surrounding areas for additional mines
- If a ship is detected, AUV should track it to gather information
- When the mission is completed, AUV should return to the base

mission description in a mathematical model

What we would like to do

mission description in a natural, structured, language

- AUV should always be safe
- Objective is to inspect areas $A_1, A_2, \ldots, A_n$
- AUV must adapt to the environment, avoiding any obstacles it encounters
- If a mine is detected, AUV should explore surrounding areas for additional mines
- If a ship is detected, AUV should track it to gather information
- When the mission is completed, AUV should return to the base

mission description in a mathematical model

"(**always** safe) **and** (**eventually** (inspect areas $A_1, A_2, \ldots, A_n$ **and**
(**if** obstacle **then** avoid) **and**
(**if** elevation change **then** adapt) **and**
(**if** indication of mines **then** explore surrounding area) **and**
(**if** indication of ship **then** track **until** identified))
followed by return to the base)"

What we would like to do

mission description in a natural, structured, language

- AUV should always be safe
- Objective is to inspect areas $A_1, A_2, \ldots, A_n$
- AUV must adapt to the environment, avoiding any obstacles it encounters
- If a mine is detected, AUV should explore surrounding areas for additional mines
- If a ship is detected, AUV should track it to gather information
- When the mission is completed, AUV should return to the base

mission description in a mathematical model

"(**always** safe) **and** (**eventually** (inspect areas $A_1, A_2, \ldots, A_n$ **and**
(**if** obstacle **then** avoid) **and**
(**if** elevation change **then** adapt) **and**
(**if** indication of mines **then** explore surrounding area) **and**
(**if** indication of ship **then** track **until** identified))
followed by return to the base)"

automatically plan motions to accomplish the mission

Motivation for Proposed Approach

- Formulating a generalized mission with a series of objectives over a time span rather than a set of waypoints with specific tasks

Motivation for Proposed Approach

- Formulating a generalized mission with a series of objectives over a time span rather than a set of waypoints with specific tasks

- Leveraging state-of-the-art motion planning to navigate through complex environments while acomplishing mission objectives

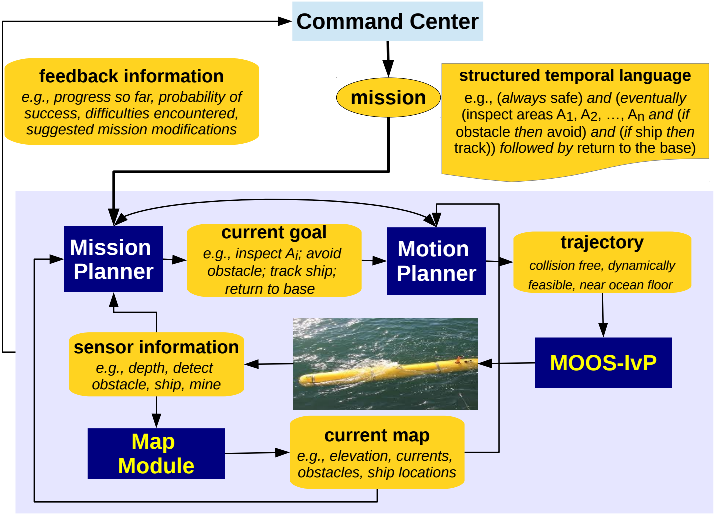
Motivation for Proposed Approach

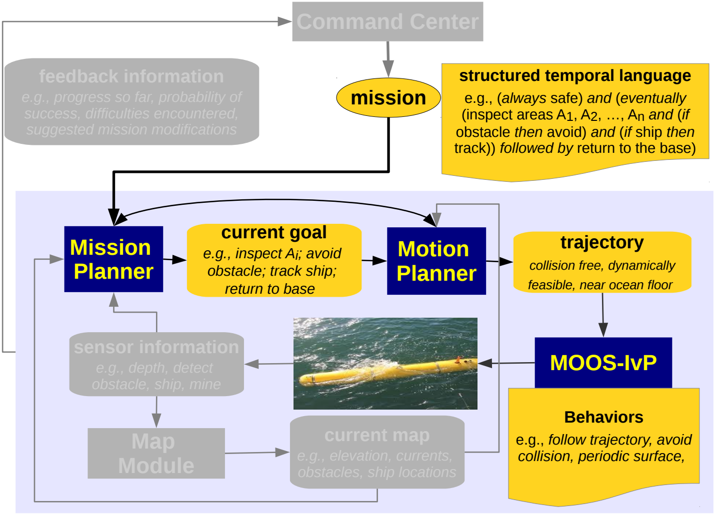
- Formulating a generalized mission with a series of objectives over a time span rather than a set of waypoints with specific tasks
- Leveraging state-of-the-art motion planning to navigate through complex environments while acomplishing mission objectives
- Having the ability to explore new areas not included in initial mission plan

Computational Challenges

- Decision-making mechanisms in response to global and local events
- Operating in confined areas and waterways close to the ocean floor
  - Varying ocean currents
  - Complex ocean-floor topography
  - Miscellaneous obstacles, e.g., wreckage, boulders, fishing nets
- Robustly adapting to changing environmental and contextual conditions
- Accounting for the underlying AUV dynamics

Overall Proposed Framework




Command Center
feedback information
e.g., progress so far, probability of success, difficulties encountered, suggested mission modifications
mission
structured temporal language
e.g., (always safe) and (eventually (inspect areas A1, A2, …, An and (if obstacle then avoid) and (if ship then track)) followed by return to the base)
Mission Planner
current goal
e.g., inspect Ai; avoid obstacle; track ship; return to base
Motion Planner
trajectory
collision free, dynamically feasible, near ocean floor
sensor information
e.g., depth, detect obstacle, ship, mine
MOOS-IvP
Behaviors
e.g., follow trajectory, avoid collision, periodic surface,
Map Module
current map
e.g., elevation, currents, obstacles, ship locations

Outline

Mission Specifications via Linear Temporal Logic (LTL)

- LTL provides an expressive mathematical model to express tasks
- LTL combines propositions Π with logical (and [$\wedge$], or [$\vee$], not [$\neg$]) and temporal operators (next [$\bigcirc$], eventually [$\Diamond$], until [$\cup$], always [$\Box$]), e.g.,
  - coverage: "search areas $A_1, \ldots, A_n$ in any order" $\Diamond A_1 \wedge \ldots \wedge \Diamond A_n$
  - sequencing: "inspect A_1, A_2, A_3 in order' $\Diamond A_1 \wedge (\Diamond A_2 \wedge (\Diamond A_3))$
  - partial ordering: "visit A_1 or A_2 before A_3 or A_4"
    $(\neg A_3 \wedge \neg A_4) \cup ((A_1 \vee A_2) \wedge \bigcirc(A_3 \vee A_4))$
  - conditions: "if ostacle detected then avoid; if moving object detected, then track until identified;"
    $\Box((obstacle \Rightarrow \bigcirc avoid) \wedge (moving\_object \Rightarrow (track \cup identified)))$

Mission Specifications via Linear Temporal Logic (LTL)

Sophisticated missions can be constructed by composing simpler ones

- Mission for inspecting offshore platform can use propositions to express status (damage or functional) of pipes, valves, anchors, anchor lines, flotation chambers
- Partial ordering can be used to prioritize the inspection of critical components
- Conditional constructs can be employed to carry out closer inspections when there is some indication of damage of a particular component
- Avoidance and persistency can ensure a safe minimum distance away from the platform components, while still being close enough to carry out inspections
- Coverage criteria can ensure that all components have been inspected

Outline

Controller Synthesis from LTL Specifications

Construct a controller that drives the robot in such a way that the resulting trajectory satisfies the LTL formula ϕ

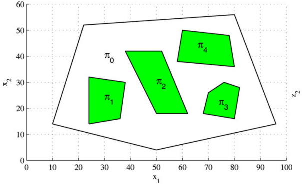
[Kress-Gazit et al., 2007–2013; Feinekos et al., 2009, 2011; Belta et al., 2008–2013; LaValle, 2011]

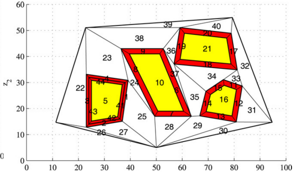
Construct a controller that drives the robot in such a way that the resulting trajectory satisfies the LTL formula ϕ

[Kress-Gazit et al., 2007–2013; Feinekos et al., 2009, 2011; Belta et al., 2008–2013; LaValle, 2011]

Decoupled Framework

- Decompose 2D environment into convex polygons, e.g., triangles





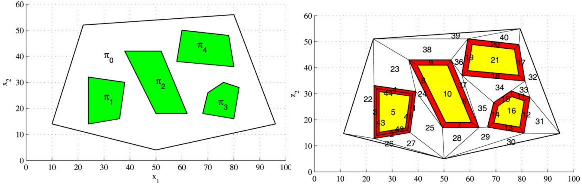
Related Work: Controller Synthesis from LTL Specifications

Construct a controller that drives the robot in such a way that the resulting trajectory satisfies the LTL formula ϕ

[Kress-Gazit et al., 2007–2013; Feinekos et al., 2009, 2011; Belta et al., 2008–2013; LaValle, 2011]

Decoupled Framework

- Decompose 2D environment into convex polygons, e.g., triangles



- Use model checking to compute a sequence of decomposition regions $\tau = \tau_1, \tau_2, \cdots$ the robot needs to visit in order to satisfy ϕ
  - task: start in π_1, and then visit π_3, π_4 in any order, and then return to π_1 while avoiding π_2 and π_3 $\quad \Box\pi_0 \wedge \Diamond(\pi_2 \wedge \Diamond(\pi_3 \wedge \Diamond(\pi_4 \wedge (\neg\pi_2\neg\pi_3) \cup \Box\pi_1)))$
  - solution: $5, 41, 1, 25, 24, 8, 10, 6, 37, 35, 14, 16, 15, 34, 18, 21, 19, 36, 38, 23, 4, 44, 5$

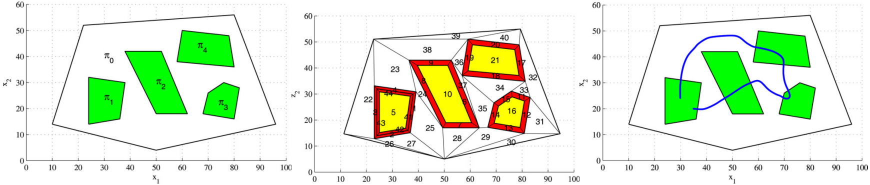
Related Work: Controller Synthesis from LTL Specifications

Construct a controller that drives the robot in such a way that the resulting trajectory satisfies the LTL formula ϕ

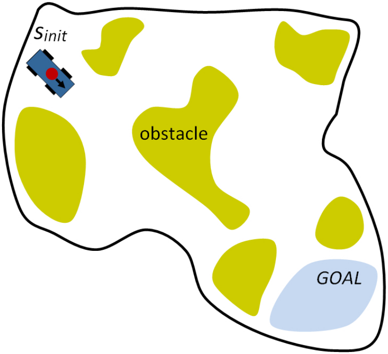
[Kress-Gazit et al., 2007–2013; Feinekos et al., 2009, 2011; Belta et al., 2008–2013; LaValle, 2011]

Decoupled Framework

- Decompose 2D environment into convex polygons, e.g., triangles

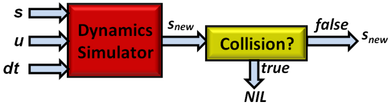


- Use model checking to compute a sequence of decomposition regions $\tau = \tau_1, \tau_2, \cdots$ the robot needs to visit in order to satisfy ϕ
  - task: start in π_1, and then visit π_3, π_4 in any order, and then return to π_1 while avoiding π_2 and π_3 $\Box\pi_0 \wedge \Diamond(\pi_2 \wedge \Diamond(\pi_3 \wedge \Diamond(\pi_4 \wedge (\neg\pi_2\neg\pi_3) \cup \Box\pi_1)))$
  - solution: $5, 41, 1, 25, 24, 8, 10, 6, 37, 35, 14, 16, 15, 34, 18, 21, 19, 36, 38, 23, 4, 44, 5$
- Use a controller, e.g., potential field, to drive the robot from one decomposition region to the next as specified in τ



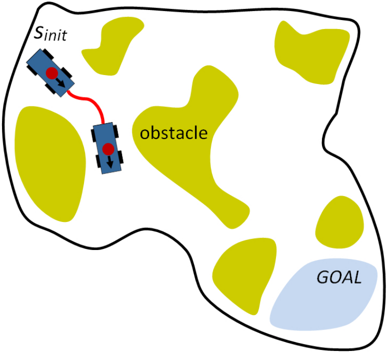
Expand a tree $\mathcal{T}$ of collision-free and dynamically-feasible motions

- select state s from which to expand tree
- sample control input u
- generate new trajectory by applying u to s



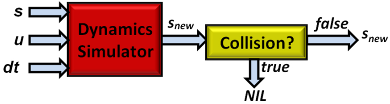
Successful motion planners: RRT, TRRT, RRT*, EST, PDST, KPIECE, SYCLOP, ...

Related Work: Sampling-based Motion Planning

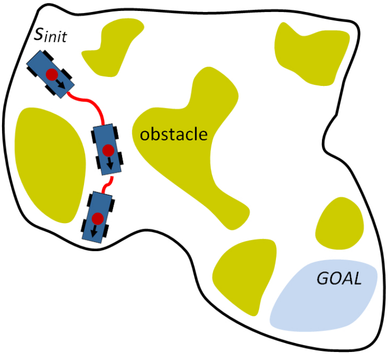


Expand a tree $\mathcal{T}$ of collision-free and dynamically-feasible motions

- select state s from which to expand tree
- sample control input u
- generate new trajectory by applying u to s

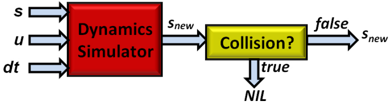


Successful motion planners: RRT, TRRT, RRT*, EST, PDST, KPIECE, SYCLOP, . . .



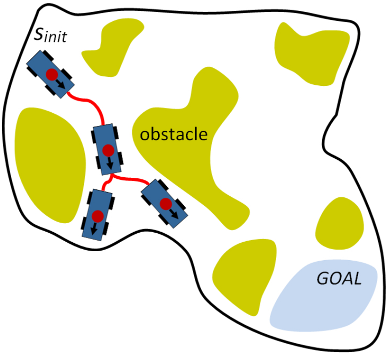
Expand a tree $\mathcal{T}$ of collision-free and dynamically-feasible motions

- select state s from which to expand tree
- sample control input u
- generate new trajectory by applying u to s



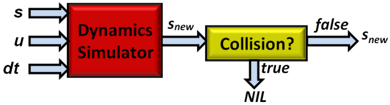
Successful motion planners: RRT, TRRT, RRT*, EST, PDST, KPIECE, SYCLOP, ...

Sampling-based Motion Planning



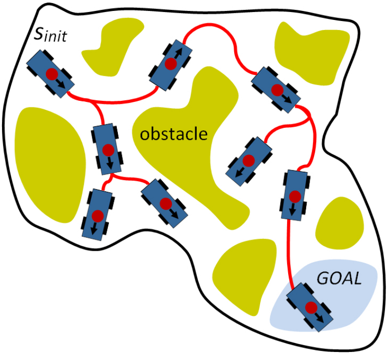
Expand a tree $\mathcal{T}$ of collision-free and dynamically-feasible motions

- select state s from which to expand tree
- sample control input u
- generate new trajectory by applying u to s



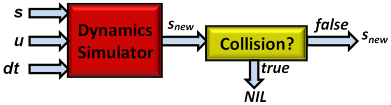
Successful motion planners: RRT, TRRT, RRT*, EST, PDST, KPIECE, SYCLOP, ...

Related Work: Sampling-based Motion Planning



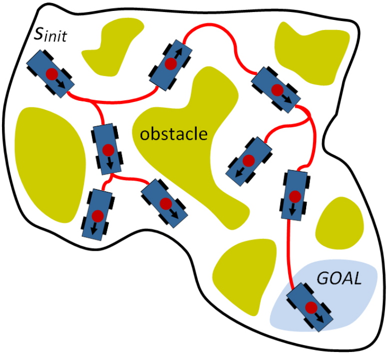
Expand a tree $\mathcal{T}$ of collision-free and dynamically-feasible motions

- select state s from which to expand tree
- sample control input u
- generate new trajectory by applying u to s



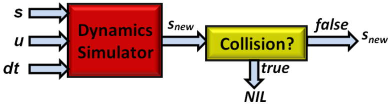
Successful motion planners: RRT, TRRT, RRT*, EST, PDST, KPIECE, SYCLOP, . . .

Related Work: Sampling-based Motion Planning



Expand a tree $\mathcal{T}$ of collision-free and dynamically-feasible motions

- select state s from which to expand tree
- sample control input u
- generate new trajectory by applying u to s

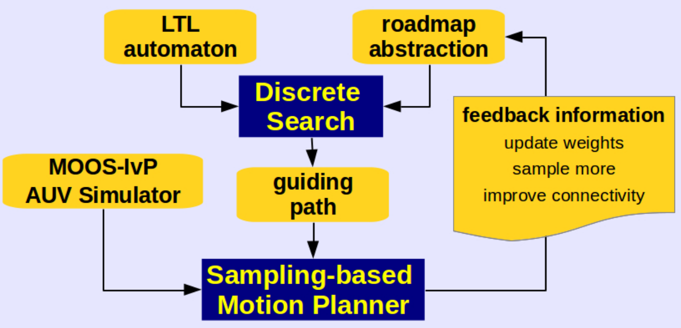


Successful motion planners: RRT, TRRT, RRT*, EST, PDST, KPIECE, SYCLOP, ...

... but sampling-based motion planning on its own cannot take into account LTL specifications

Outline

Proposed Approach: Coupled Mission and Motion Planning



discrete layer: guide motion planning

continuous layer: expand tree of feasible motions

interplay: update guide to reflect motion-planning progress

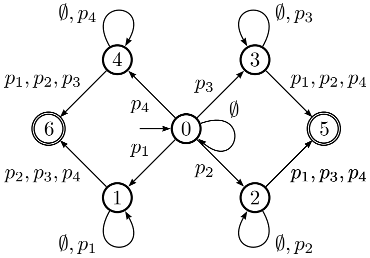
- Builds upon coupled framework proposed by [Plaku, Kavraki, Vardi, TRO 2010; Plaku IROS 2011; Plaku TAROS 2012]
- Generally applicable to high-dimensional systems with nonlinear dynamics
- Works in 3D environments

Roadmap Abstraction in Configuration Space

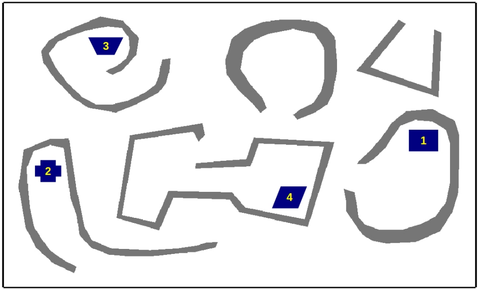
- Roadmap captures connectivity of the free configuration space
- Roadmap provides simplified abstraction layer

  configuration space ignores dynamics, so easier to plan
- Used to facilitate motion planning in the full state space, taking AUV dynamics into account

task: "visit any two of the regions 1,2,3,4"



converted to LTL automaton

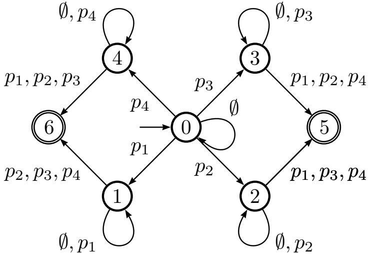


Roadmap Abstraction in Configuration Space

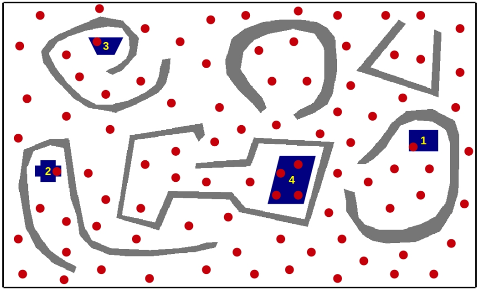
- Roadmap captures connectivity of the free configuration space
- Roadmap provides simplified abstraction layer

  configuration space ignores dynamics, so easier to plan
- Used to facilitate motion planning in the full state space, taking AUV dynamics into account

task: "visit any two of the regions 1,2,3,4"



converted to LTL automaton



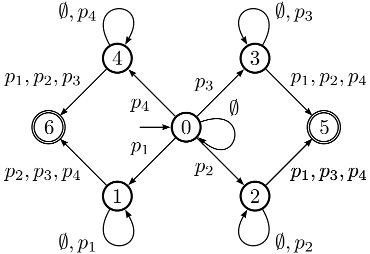
- Sample collision-free configurations

Roadmap Abstraction in Configuration Space

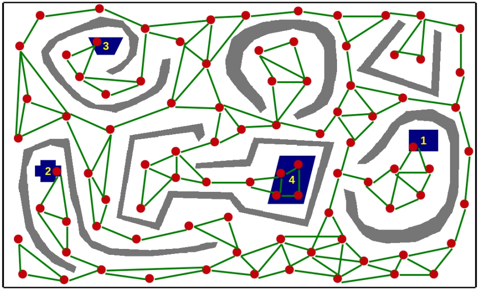
- Roadmap captures connectivity of the free configuration space
- Roadmap provides simplified abstraction layer

  configuration space ignores dynamics, so easier to plan
- Used to facilitate motion planning in the full state space, taking AUV dynamics into account

task: "visit any two of the regions 1,2,3,4"



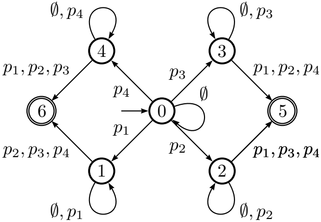
converted to LTL automaton

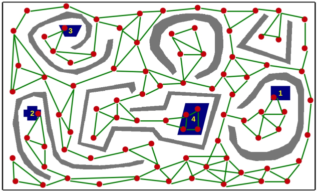


- Sample collision-free configurations
- Connect neighboring configurations

Guiding the Search: Combining Roadmap with LTL Automaton

Guiding path $\sigma = [\langle z_1, c_1 \rangle, \ldots, \langle z_n, c_n \rangle]$ connects initial pair $\langle z_{\text{init}}, c_{\text{init}} \rangle$ to an accepting automaton state so that LTL formula is satisfied



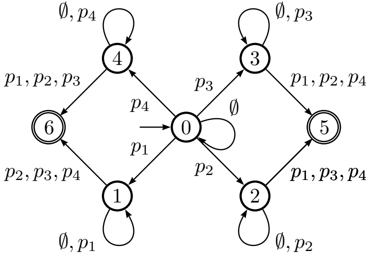


Guiding path provides an approximate path of how sampling-based motion-planning should expand the motion tree to satisfy LTL formula

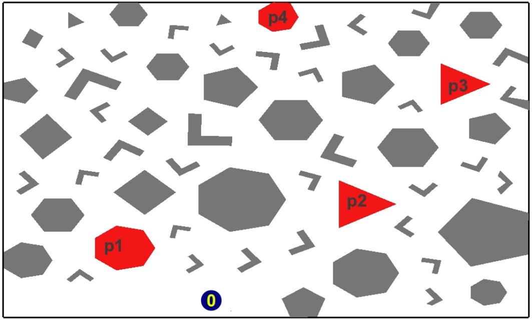
Search conducted over graph $RA = (V_{RA}, E_{RA})$

RA obtained by combining implicitly roadmap RM with automaton $\mathcal{A}$

Any graph search over RA can be used, e.g., DFS, BFS, Dijkstra, A* task: "visit any two of the regions p1, p2, p3, p4"

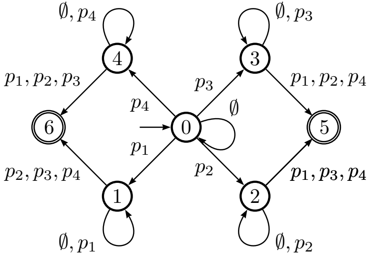


converted to LTL automaton

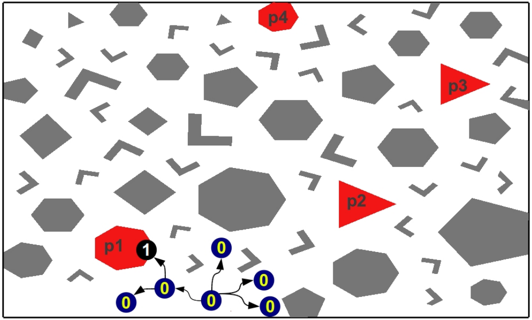


Expanding the Tree of Motions

task: "visit any two of the regions p1, p2, p3, p4"

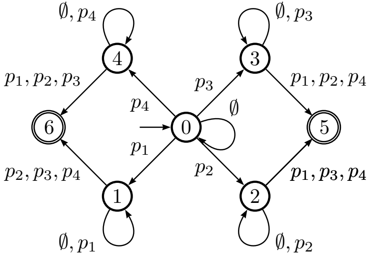


converted to LTL automaton

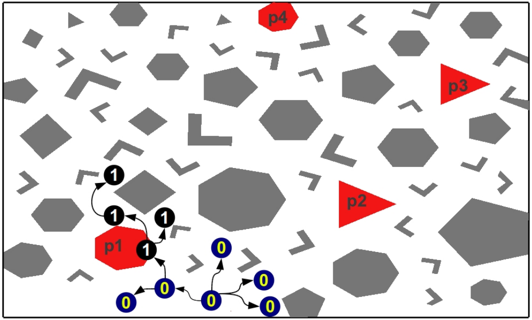


Expanding the Tree of Motions

task: “visit any two of the regions p1, p2, p3, p4”

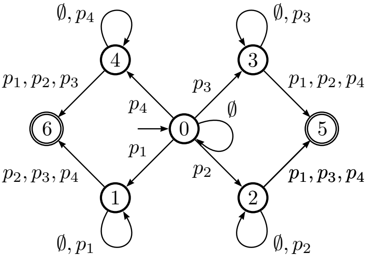


converted to LTL automaton

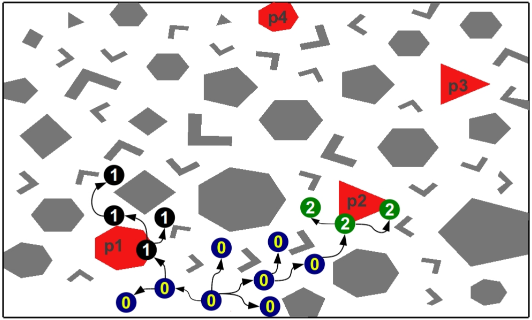


Expanding the Tree of Motions

task: "visit any two of the regions p1, p2, p3, p4"

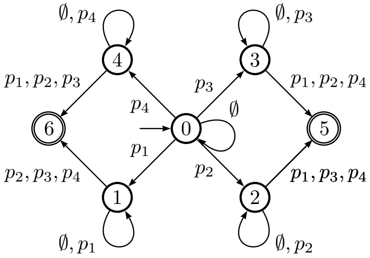


converted to LTL automaton

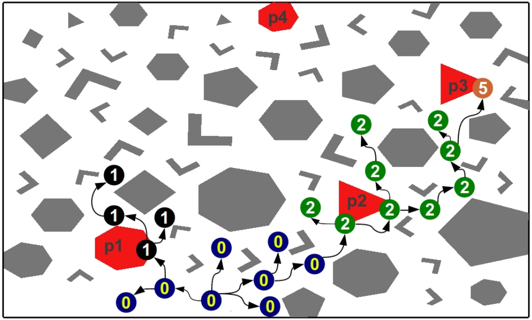


Expanding the Tree of Motions

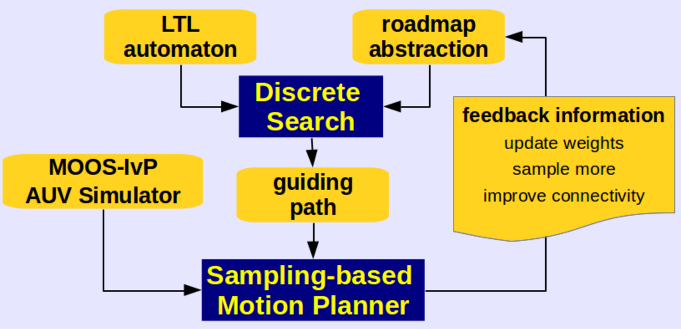
task: “visit any two of the regions p1, p2, p3, p4”



converted to LTL automaton



Interplay Among the Layers



discrete layer: guide motion planning

continuous layer: expand tree of feasible motions

interplay: update guide to reflect motion-planning progress

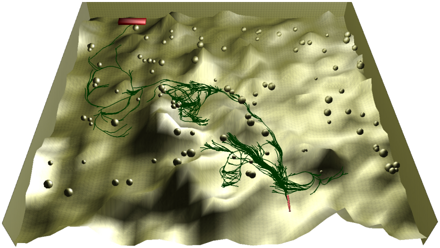
Outline

Simulation Environment

- Based on MOOS-IvP (uSimMarine) (Benjamin, Schmidt, Newman, Leonard: J. Field Robotics 2010)
- Models vehicle dynamics
- Takes into account drift caused by ocean currents
- Operates in 3D environments

For planning purposes, abstracted as

$$s_{\text{new}} \leftarrow \textsc{Simulator}(s, u, \text{drift}, dt)$$

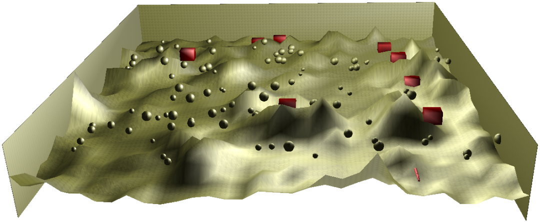


- Simulated ocean floor created by adding random peaks
- Height at grid cell based on distance to closest peak
- Heightmap converted to 3D triangular mesh
- AUV needs to operate close to ocean floor

Provides initial validation

Some Examples of Mission Specifications via LTL

- Each area of interest A_i defines a proposition π_{A_i}
- $\text{HOLDS}_{\pi_{A_i}}(s)$ is true iff state s places AUV in A_i



Compute a collision-free and dynamically-feasible trajectory ζ which satisfies

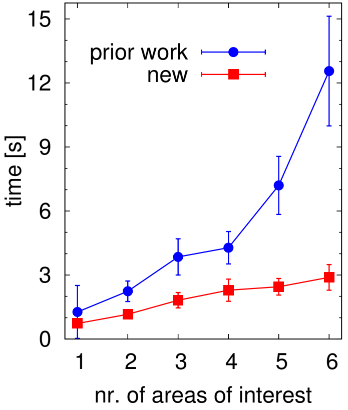
1. Inspect all:

$$\phi_1 = \bigwedge_{i=1}^{n} \Diamond \pi_{A_i}$$

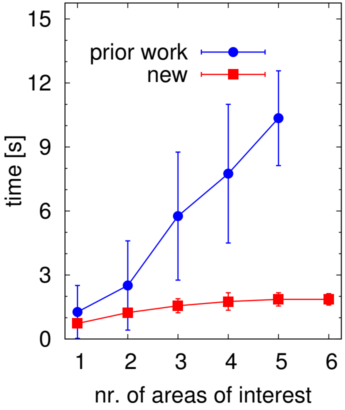
2. Visit $A_1, A_2, \ldots, A_n$ in succession, i.e.,

$$\phi_2 = \beta \cup (\pi_{A_1} \wedge ((\pi_{A_1} \vee \beta) \cup (\pi_{A_2} \wedge (\ldots(\pi_{A_{n-1}} \vee \beta) \cup \pi_{A_n})))),$$

where $\beta = \wedge_{i=1}^{n} \neg \pi_i$



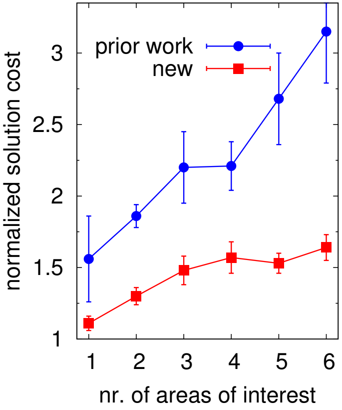
(a) task ϕ_1: "all"



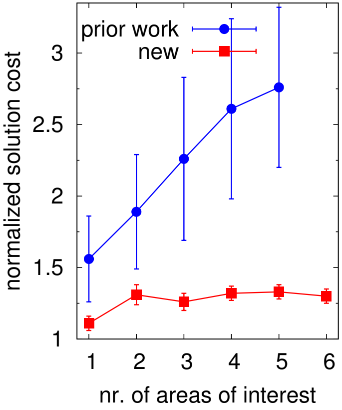
(b) task ϕ_2: "sequencing"

- prior work: Plaku [TAROS 2012] – no roadmap abstraction
- new work: Plaku and McMahon [TAROS 2013] – with roadmap abstraction

Results: Trajectory Cost



(a) task ϕ_1: “all”



(b) task ϕ_2: “sequencing”

path $\leftarrow$ shortest path in abstract graph formed by roadmap and LTL automaton
path is in configuration space, ignores dynamics

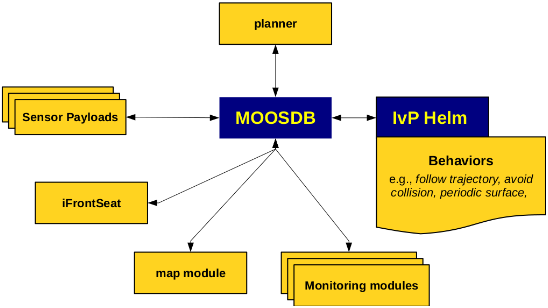
traj $\leftarrow$ solution trajectory obtained in full state space

normalize $\leftarrow ||traj||/||path||$

Outline

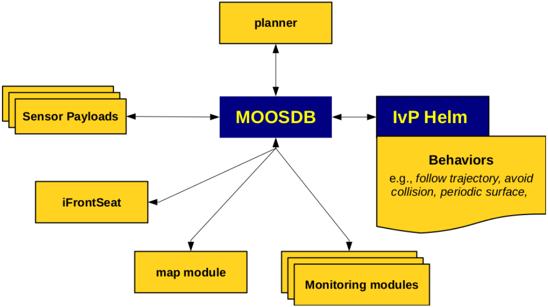
High Level Planning in the MOOS-IvP Framework

Use the high level planner to generate feasible trajectories that satisfy the LTL formula while employing MOOS-IvP for reactionary behaviors



High Level Planning in the MOOS-IvP Framework

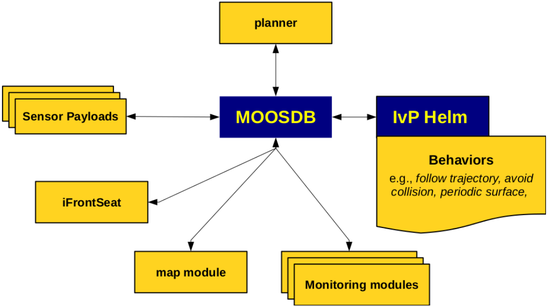
Use the high level planner to generate feasible trajectories that satisfy the LTL formula while employing MOOS-IvP for reactionary behaviors



- Create an IvP behavior that accepts planned trajectories in state space(*e.g., a dynamic set of configurations, X,Y,Heading,Speed,Depth*)

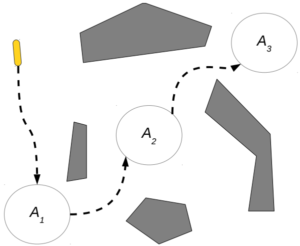
High Level Planning in the MOOS-IvP Framework

Use the high level planner to generate feasible trajectories that satisfy the LTL formula while employing MOOS-IvP for reactionary behaviors



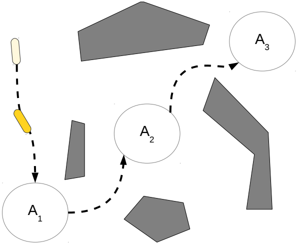
- Create an IvP behavior that accepts planned trajectories in state space(*e.g., a dynamic set of configurations, X,Y,Heading,Speed,Depth*)
- Use existing IvP behaviors to help handle dramatic changes to the environment (*e.g., Avoid Collision*)

Using MOOS-IvP and the LTL framework to satisfy the statement (**always** safe) **and** (**eventualy** inspect areas A_1, A_2, A_3)



- The inital trajectory is generated by the LTL planner

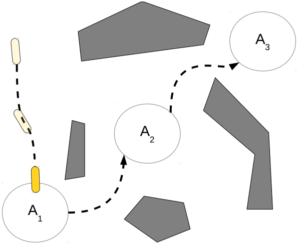
Using MOOS-IvP and the LTL framework to satisfy the statement (**always** safe) **and** (**eventualy** inspect areas A_1, A_2, A_3)



- The inital trajectory is generated by the LTL planner

Integrating LTL Planner with MOOS-IvP

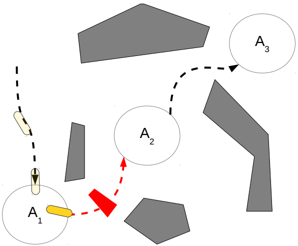
Using MOOS-IvP and the LTL framework to satisfy the statement (**always** safe) **and** (**eventualy** inspect areas A_1, A_2, A_3)



- The inital trajectory is generated by the LTL planner

Integrating LTL Planner with MOOS-IvP

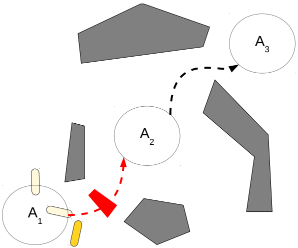
Using MOOS-IvP and the LTL framework to satisfy the statement
(**always** safe) **and** (**eventualy** inspect areas A_1, A_2, A_3)



- The inital trajectory is generated by the LTL planner
- A unkown obstacle is suddenly discovered in the planned path

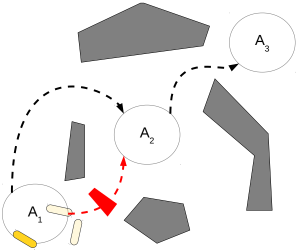
Integrating LTL Planner with MOOS-IvP

Using MOOS-IvP and the LTL framework to satisfy the statement (**always** safe) **and** (**eventualy** inspect areas A_1, A_2, A_3)



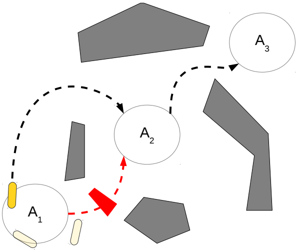
- The inital trajectory is generated by the LTL planner
- A unkown obstacle is suddenly discovered in the planned path
- Reactive behaviors in MOOS-IvP prevent the AUV from colliding with the obstacle

Using MOOS-IvP and the LTL framework to satisfy the statement
(**always** safe) **and** (**eventualy** inspect areas A_1, A_2, A_3)



- The inital trajectory is generated by the LTL planner
- A unkown obstacle is suddenly discovered in the planned path
- Reactive behaviors in MOOS-IvP prevent the AUV from colliding with the obstacle
- The LTL planner generates a new trajectory for the AUV to follow

Using MOOS-IvP and the LTL framework to satisfy the statement
(**always** safe) **and** (**eventualy** inspect areas A_1, A_2, A_3)



- The inital trajectory is generated by the LTL planner
- A unkown obstacle is suddenly discovered in the planned path
- Reactive behaviors in MOOS-IvP prevent the AUV from colliding with the obstacle
- The LTL planner generates a new trajectory for the AUV to follow

Future Work

- IvP behaviors to follow trajectories within configuration space
- MOOS bathymetric mapping application
- Incorporate ocean models to plan with predicted currents
- Re-planing framework that incorporates state information that exists within the MOOSDB
- Perform experimental evalution using Bluefin 21" vehicle

Summary

Framework couples

- Discrete planning to take into account LTL specifications with
- Sampling-based motion planning to handle motion dynamics and obstacles
- Reactionary behaviors to handle unforseen dramatic changes in the environment

Roadmap abstraction combined with LTL automaton effectively guides search in the continuous state space

Experiments with accurate AUV simulators provide promising initial validation