Symbolic Planning for Dynamic Robots

Laya Shamgah
Advisor: Dr. Karimoddini
TECHLAV Project

- Testing, Evaluation and Control of Heterogeneous Large-scale Autonomous systems of Vehicles (TECHLAV)

- Thrust 1: Modeling, Analysis and Control of Large-scale Autonomous Vehicles (MACLAV)

- Task 1-5: Hierarchical Hybrid Cooperative Control of LSASV
### TECHLAV Project

<table>
<thead>
<tr>
<th><strong>Objective</strong></th>
<th><strong>Impact</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>Developing capable and scalable models for autonomous collaborative, robust and distributed decision-making, group coordination, planning, and tasking through effective interaction with human operators.</td>
<td>Reaching higher levels of autonomy and teaming of multi-agent systems by using agents which are able to accomplish assigned missions autonomously and have the capability of autonomous collaboration with other teammates and human operators.</td>
</tr>
</tbody>
</table>
Relation to TECHLAV

Task 1-5: Hierarchical Hybrid Cooperative Control of LSASV

Hierarchical Control

- Supervisory
- Planning
- Real-time control system

Symbolic planning facilitates planning and coordination in a hierarchical control structure.

Rules and logics, ...

Actuators, ...
Outline

1. Robot Motion Planning and Control
2. Task Specification
3. Temporal Logic
4. Symbolic Planning
5. Future Work
# Mobile Robots

## Components
- mechanical elements (wheels and gears)
- electromechanical devices (motors, clutches and brakes)
- digital circuits (processors and smart sensors)
- software programs (embedded controllers)

## Constraints
- mechanical constraints (e.g., a car-like robot cannot move sideways)
- limited energy resources, and computation, sensing, and communication capabilities.

## Complexities
- Environment is cluttered with possibly moving and shape changing obstacles.
- Their objectives can change over time, such as in the case of appearing and disappearing targets.
New Task Requirements for Robots

- Converging to a desired operating point while always staying within a safe set
- Executing sequenced tasks
- Reaching certain areas and visiting certain areas infinitely often
- Avoiding obstacles
Robot Motion Planning and Control

Problem
- automatic construction of robot control strategies from task specifications given in high-level, human-like language

Challenge
- the development of frameworks for control design which is:
  - computationally efficient
  - allowing for systematic, provably correct, control design
  - accommodating both the robot constraints and the complexity of the environment
  - allowing for expressive human-like task specifications
Mission requirements:
• converging to a desired operating point while always staying within a safe set
• executing sequenced tasks
• reaching certain areas and visiting certain areas infinitely often
• avoiding obstacles

How to formally describe the task?
Temporal logic
Why Temporal Logic?

- A formal high level language to describe a complex mission
- A wider class of properties than safety and stability
- Having well defined syntax and semantics, which can be easily used to specify complex behavior
Introduction to Temporal Logic

Temporal logic:
Any system of rules and symbolism for representing, and reasoning about, propositions qualified in terms of time.

Classical logic:
- “It is Monday”

Temporal logic
- "I am always hungry“
- "I will eventually be hungry“
- "I will be hungry until I eat something"
## Temporal Operators

<table>
<thead>
<tr>
<th>Operators</th>
<th>Definition</th>
<th>Diagram</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\circ \varphi$</td>
<td>$\varphi$ in true in the next moment of time</td>
<td>![Diagram for $\circ \varphi$]</td>
</tr>
<tr>
<td>$\Box \varphi$</td>
<td>$\varphi$ is true in all future moments</td>
<td>![Diagram for $\Box \varphi$]</td>
</tr>
<tr>
<td>$\Diamond \varphi$</td>
<td>$\varphi$ is true in some future moment</td>
<td>![Diagram for $\Diamond \varphi$]</td>
</tr>
<tr>
<td>$\varphi u \psi$</td>
<td>$\varphi$ is true until $\psi$ is true</td>
<td>![Diagram for $\varphi u \psi$]</td>
</tr>
</tbody>
</table>
Temporal Syntax

$\varphi ::= p \mid \neg \varphi \mid \varphi \land \varphi \mid \varphi \lor \varphi \mid \varphi \rightarrow \varphi$

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P$</td>
<td>an arbitrary atomic proposition</td>
</tr>
<tr>
<td>$\neg$</td>
<td>Negation (complement)</td>
</tr>
<tr>
<td>$\land$</td>
<td>conjunction</td>
</tr>
<tr>
<td>$\lor$</td>
<td>disjunction</td>
</tr>
<tr>
<td>$\rightarrow$</td>
<td>implication</td>
</tr>
</tbody>
</table>

Example:

$(p \land \neg q) \rightarrow r$
## Temporal Logic-Examples

<table>
<thead>
<tr>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>□((¬passport ∨ ¬ticket) → ○¬board_flight)</td>
</tr>
<tr>
<td>□(requested → ◊received)</td>
</tr>
<tr>
<td>□(received → ○processed)</td>
</tr>
<tr>
<td>□(processed → ◊□done)</td>
</tr>
</tbody>
</table>
Temporal Semantics

- A tool for specification, formal analysis, and verification of the executions of computer programs and systems

<table>
<thead>
<tr>
<th>$\mathcal{M} = \langle \mathbb{N}, I \rangle$</th>
<th>a discrete, linear model of time</th>
</tr>
</thead>
<tbody>
<tr>
<td>$I : \mathbb{N} \rightarrow 2^\Sigma$</td>
<td>maps each natural number to a set of propositions</td>
</tr>
<tr>
<td>$\models : (\mathcal{M} \times \mathbb{N} \times \text{FORM}) \rightarrow {\text{true, false}}$</td>
<td>Satisfaction relation</td>
</tr>
</tbody>
</table>

$\langle \mathcal{M}, i \rangle \models p \iff p \in I(i)$ \hspace{1em} (for $p \in \Sigma$)
LTL for Robot Task Specifications

- **Coverage:** eventually visit all regions
- **Sequencing:** visit P2 before you go to P3
- **Avoidance:** until you go to P2 avoid P1 and P3

**Examples:**

1. “visit rooms 1, 2, 3 while avoiding corridor 1!”
   \[ \varphi = \Box \neg (\text{corridor1}) \land \Box (\text{room1}) \land \Box (\text{room2}) \land \Box (\text{room3}) \]

2. “if the light is on, visit rooms 1 and 2 infinitely often!”
   \[ \varphi = \Box (\text{lightOn}) \rightarrow (\Box \Diamond (\text{room1}) \land \Box \Diamond (\text{room2})) \]

3. “if you are in room 3 and Mike is there, beep!”
   \[ \varphi = \Box (\text{room3}) \land (\text{SeeMike}) \rightarrow (\text{beep}) \]
Symbolic Planning

\[ \varphi = \square ((\text{room3}) \land (\text{SeeMike}) \rightarrow (\text{beep})) \]

**Task**

**Challenge:**
Given task \( \varphi \) and a dynamical model of the robot construct controllers such that:
Robot | Controller \( \vdash \varphi \)

**Abstraction**

\[ \dot{x}(t) = u(t) \]
\[ x(t) \in P \subset \mathbb{R}^2 \]
\[ u(t) \in U \subset \mathbb{R}^2 \]
Control Synthesis

Verification

Planning

\[
\varphi \\
S \models \varphi ? \\
\text{Control system } S \\
\text{R-DES} \rightarrow \text{R}
\]

\[
\varphi \\
\varphi \text{-DES} \\
\text{Control System?} \\
\text{R-DES} \rightarrow \text{R}
\]
Applying the DES Supervisor to the Robot

Challenge:
- DES supervisor tells the robot to go from P1 to P2.
- The planner has no idea about the status of robot.

Question:
- How to map the high-level task into the continuous controller?
Bisimulation Equivalency

- We have to couple equivalency to specification.

Language equivalency

- Preserves LTL properties
- Continuous system and DES generate the same trajectories

Bisimulation equivalency

- Preserves LTL properties
- Continuous system and DES are equivalent in all states (stronger)
Symbolic Planning

\[ \varphi = \Box (\text{room3}) \land (\text{SeeMike} \rightarrow \text{beep}) \]

if (DES supervisor \models \varphi) then
(hybrid controller \mid \mid \text{robot} \models \varphi)

\[ \dot{x}(t) = u(t) \]
\[ x(t) \in P \subset \mathbb{R}^2 \]
\[ u(t) \in U \subset \mathbb{R}^2 \]
Future Work

Symbolic Planning

Avoidance Problem

?
Future Work

Avoidance problem

Obstacle avoidance

Collision avoidance

Reach-avoidance
Future Work

Reach-avoidance

**Graph theory**
Partitions the graph into regions that are defended and cleared by a team of pursuers to solve simplified path-planning.

**Discrete games**
Visibility-based pursuit-evasion games: a group of searchers attempt to bring an evader into their field of view, through cooperative coverage of regions.

**Path planning with obstacle**
Use the possible future positions of the obstacles as static obstacles, allowing using static planning methods (Certain applications with simple configurations of moving adversarial obstacles,).

**model-predictive control**
Predicts opponent actions so that optimization can be performed for the controlled agents with respect to the assumed opponent behavior.

**Differential game**
Winning strategies for the opposing agents can be viewed as the solution to a zero-sum differential game. Simple tasks.
1. Formal description of tasks
2. Efficient computation
3. Combining reaching problem with avoiding problem
4. Targeting more complex tasks (sequencing and ... )
References


• “MDP Optimal Control under Temporal Logic Constraints”, Xu Chu Ding Stephen L. Smith Calin Belta Daniela Rus,CDC 2011.

• “Automatic Synthesis of Robust Embedded Control Software”, Tichakorn Wongpiromsarn, Ufuk Topcu and Richard M. Murray.


Thank you 😊

Question?