Case Study: Personal Transportation System

Masahiro Ono, Peng Yu

October 26th, 2011
Reminder

• 16.413 Project Part 1:
  – Out last Wednesday.
  – Due Nov, 14th.

• Mid-term:
  – Monday Oct, 31st, Halloween.
  – 1 letter-size help sheet, print or hand-written.
  – 9:30am, Rm 33-419.
  – 85 minutes.
Motivation

• 50 years later, every household will have a personal aircraft (VTOL).
Motivation

• However, flying aircraft is not easy:
  – Single Engine: 3 months
  – Multiengine Commercial: 6 months
  – Helicopter: 3 months

• Create a highly automated vehicle:
  – Provides point-to-point transportation like a taxi
  – Must be robust to uncertainty
  – Taxi driver!
Demo

• The Personal Transportation System with X-Plane Simulation.
System Architecture

Dynamic Coaching

Dialog Management
- Provides natural language interface

Collaborative Diagnosis
- Resolves plan infeasibility

Kirk
- Makes high-level decisions

p-Sulu
- Plans path within risk bounds

User
- Natural Language

Stanford/Boeing

PTS
- Control Commands

MIT
- Massachussetts Institute of Technology
System Architecture

Dynamic Coaching

Dialog Management
- Provides natural language interface
- Resolves plan infeasibility
- Plans path within risk bounds

Collaborative Diagnosis
- Makes high-level decisions

Kirk

p-Sulu
- Makes high-level decisions
- Plans path within risk bounds

User
- Natural Language

PTS
- Control Commands

Stanford/Boeing
Generate Temporal Plan

• Convert user requirements into temporal plan.
  – I want to go to the Boeing company.
  – I want to be there in 3 minutes.
  – I want to use Harvey Field as backup landing sites.
  – I want to stop at Leisureland if possible.
Generate Temporal Plan

- Convert user requirements into temporal plan.
  - I want to go to the Boeing company.
  - I want to be there in 3 minutes.
  - I want to use Harvey Field as backup landing sites.
  - I want to stop at Leisureland if possible.
Generate Temporal Plan

- I want to go to the Boeing company.
- I want to be there in 3 minutes.
- I want to use Harvey Field as Backup landing sites.
- I want to stop at Leisureland if possible.
Generate Temporal Plan

- Convert user requirements into temporal plan.
- Estimate the flight durations.
Generate Temporal Plan

- Convert user requirements into temporal plan.
- Estimate the flight durations.
- Add user preferences.

![Diagram]

- Current Location
- Fly: [2, 2] Reward: 10
- Leisureland
- Fly: [4, 4] Reward: 10
- Boeing
- Company
- [0, 3] Reward: 3
- [5, 5] Reward: 5
- [0, 3] Reward: 3
Temporal Plan Network (Kim, Williams and Abramson, 2001)

- Augmented from Simple Temporal Networks.
  - Addition of decision nodes.
  - Rewards/costs.
  - Symbolic constraints.
Solve a TPN

- To find the most preferred/least cost plan.
  - Generate the best candidate.
  - Check temporal consistency.
  - Return solution (if candidate consistent) or start over (generate the next best candidate).
Solve a TPN

• To find the most preferred/least cost plan.
  – Generate the best candidate.
  – Check temporal consistency.
  – Return solution (if candidate consistent) or start over (generate the next best candidate).

- Current Location
- Boeing Company
- Leisureland

[0, 3] Reward: 3
[0, 3] Reward: 10
[5, 5] Reward: 10
[2, 2] Reward: 5
[4, 4] Reward: 10

Fly:
Leisureland
Boeing
Company
Current
Location

15
Solve a TPN

- To find the most preferred/least cost plan.
  - Generate the best candidate.
  - Check temporal consistency.
  - Return solution (if candidate consistent) or start over (generate the next best candidate).
Solve a TPN

- To find the most preferred/least cost plan.
  - Generate the best candidate.
  - Check temporal consistency. Not consistent!
  - Return solution (if candidate consistent) or start over (generate the next best candidate).
Solve a TPN

• To find the most preferred/least cost plan.
  – Generate the best candidate.
  – Check temporal consistency.
  – Return solution (if candidate consistent) or start over (generate the next best candidate).
Solve a TPN

- To find the most preferred/least cost plan.
  - Generate the best candidate.
  - Check temporal consistency. Not consistent!
  - Return solution (if candidate consistent) or start over (generate the next best candidate).

```
Fly: [5, 5]  
Reward: 10
```

```
Fly: [2, 2]  
Reward: 10
```

```
Fly: [0, 3]  
Reward: 3
```

```
Fly: [4, 4]  
Reward: 10
```

---

Massachusetts Institute of Technology
What if no solution exists…

• Tell the user I cannot find a solution.
• Let the user figure out the problem and input a new set of requirements.

• OR

• Diagnose the over-constrained plan and find a relaxation for the user.
  – “If you relax your constraints or fly faster, I can find a feasible plan for you.”
System Architecture

Dynamic Coaching

Dialog Management
- Provides natural language interface

Collaborative Diagnosis
- Resolves plan infeasibility
- Makes high-level decisions

Kirk

p-Sulu
- Plans path within risk bounds

User

Natural Language

Stanford/Boeing

PTS

Control Commands

MIT

Massachusetts Institute of Technology
In the PTS Scenario

PTS

You cannot get there in 3 minutes but you can get there in 6 minutes.

User

Collaborative Diagnosis:

– Generate plan.
– Detect and diagnose conflicts.
– Present diagnoses and repair options to user.
Collaborative Diagnosis - Introduction

- Definition
  - An interface between the computer and the user.
Collaborative Diagnosis - Introduction

• Definition
  – An interface between the computer and the user.

• Objective
  – Help the user resolve infeasible plans.

Infeasible Plan

Feasible Plan

"You can relax your time constraint to 6 minutes to restore plan consistency"
Challenge and Key Idea

• Challenge: Too many options to take.
• Key Idea: Implement the diagnosis concepts and reduce the size of results by intelligently pruning meaningless options.
  – Current-WA96 $\rightarrow$ \{IN, OUT\}.
  – WA96-Boeing $\rightarrow$ \{IN, OUT\}.
  – Current-Boeing $\rightarrow$ \{IN, OUT\}.

One of the conflicting episodes should be relaxed......
Working Principle

Why is the plan infeasible?

How to repair the plan?

What is the best way to repair?

Identify the Cause of Failure

Generate minimal perturbations to the goals

Present the user with possible options
Working Principle

Why is the plan infeasible?

How to repair the plan?

What is the best way to repair?

Identify the Cause of Failure

Generate minimal perturbations to the goals

Present the user with possible options
Identify Cause of Failure

Why is the plan infeasible?

We employed Conflict-directed A* algorithm to find and resolve the conflicts that cause inconsistency.

The user constraint is in conflict with the flight duration......
Working Principle

Why is the plan infeasible?

How to repair the plan?

What is the best way to repair?

Identify the Cause of Failure

Generate minimal perturbations to the goals

Present the user with possible options
Generate Possible Options

How to repair the plan?

First, we resolve the conflicts by removing constraints (assign “OUT”).

There are two solutions for the conflict: WA96-Boeing = OUT and Constraint = OUT.
How to repair the plan?

Second, we calculate the minimal relaxation for the removed constraints.

WA96-Boeing and Constraint can be relaxed to [1, 1] and [0, 6].
Working Principle

Why is the plan infeasible?

How to repair the plan?

What is the best way to repair?

Identify the Cause of Failure

Generate minimal perturbations to the goals

Present the user with possible options
What is the best way to repair?

We present possible options to the user and let the user decide if they want to execute.

I found two options for you, which one do you prefer?

“Relax your time constraint to 6 minutes”

“Fly from Leisureland to the Boeing Company in 1 minute”
Limit

- Not efficient enough for real world problems (> 1000 episodes).

<table>
<thead>
<tr>
<th>Diagnosis Algorithm</th>
<th>Current Scenario</th>
<th>Future Scenario</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td># of Constraints</td>
<td># of Constraints</td>
</tr>
<tr>
<td></td>
<td>15</td>
<td>1000</td>
</tr>
</tbody>
</table>
System Architecture

- **Dynamic Coaching**
  - Provides natural language interface
  - Resolves plan infeasibility
  - Makes high-level decisions
  - Plans path within risk bounds

- **Dialog Management**
  - Natural Language

- **Collaborative Diagnosis**

- **Kirk**

- **p-Sulu**

- **User**
  - PTS
  - Control Commands

- **Stanford/Boeing**
Sample PTS Scenario

The passenger of the PAV wants to:
- go from Provincetown to Bedford within 60 minutes
- go through a scenic area and remain there between 5 and 10 minutes
- limit the risk of penetrating the NFZ or the storm to 0.001%
Three types of constraints

The passenger of the PAV wants to:

• go from **Provincetown** to **Bedford** within 60 minutes
• go through a **scenic area** and remain there between 5 and 10 minutes
• limit the risk of penetrating the NFZ or the storm to 0.001%

State constraints
Three types of constraints

The passenger of the PAV wants to:

- go from **Provincetown** to **Bedford** within 60 minutes
- go through a **scenic area** and remain there between 5 and 10 minutes
- limit the risk of penetrating the NFZ or the storm to 0.001%

**State constraints**

**Temporal constraints**
Three types of constraints

The passenger of the PAV wants to:

- go from **Provincetown** to **Bedford** within **60 minutes**
- go through a **scenic area** and remain there between **5** and **10** minutes
- limit the risk of penetrating the NFZ or the storm to **0.001%**

**State constraints**

**Temporal constraints**

**Chance constraints**
Three required capabilities

The passenger of the PAV wants to:

- go from Provincetown to Bedford within 60 minutes
- go through a scenic area and remain there between 5 and 10 minutes
- limit the risk of penetrating the NFZ or the storm to 0.001%

State constraints

Temporal constraints

Chance constraints

- Goal-directed planning
- Planning in continuous domain
- Risk-sensitive planning

Chance constraints:

\[ c_1 : \delta = 1\% \]
\[ c_2 : \delta = 0.0001\% \]
p-Sulu RH

**VERY** roughly speaking...

\[ p\text{-Sulu RH} = \text{probabilistic receding horizon Sulu} \]

\[ \text{p-Sulu RH} = \text{Iterative Risk Allocation} + \text{Receding horizon control} \]

Lieutenant Hikaru Sulu, from *Star Trek*
pSulu RH

VERY roughly speaking...

p-Sulu RH = Iterative Risk Allocation + Receding horizon control
Optimal control Under Stochastic Uncertainty

- Exogenous disturbance
- State estimation error

Risk of constraint violation

Nominal Path

$P(x)$

$t = 1$

$t = 2$
Example: Race Car Path Planning

- A race car driver wants to go from the start to the goal as fast as possible
- Crashing into the wall may kill the driver
- Actual path may differ from the planned path due to uncertainty
Example: Race Car Path Planning

**Problem**

Find the fastest path to the goal, while limiting the probability of crash throughout the race to 0.1%

- Cannot guarantee 100% safety
- Driver wants a probabilistic guarantee:
  \[ P(\text{crash}) < 0.1\% \]
  – Chance constraint
Example: Race Car Path Planning

Problem

Find the fastest path to the goal, while limiting the probability of crash throughout the race to 0.1%

• Approach: set safety margin that guarantees the specified risk bound from start to the goal
Optimization of Safety Margin

Uniform width

Non-uniform width

Goal
Walls
Safety margin
Start

Goal
Walls
Safety margin
Start

Longer path

Shorter path
Key Idea - Risk Allocation

- Taking a risk at the corner results in a shorter path than taking the same amount of risk at the straightaway.

- **Sensitivity** of path length to risk is higher at the corner.

- **Risk Allocation**
  - Need to optimize the allocation of risk to time steps and constraints.

Corner
Narrow safety margin = higher risk

Straightaway
Wide safety margin = lower risk
Iterative Risk Allocation (IRA) Algorithm

- Starts from a suboptimal risk allocation
- Improves the risk allocation by iterations
Iterative Risk Allocation Algorithm

**Algorithm IRA**

1. Initialize with arbitrary risk allocation
2. Loop
3. Compute the best available path given the current risk allocation
4. Decrease the risk where the constraint is inactive
5. Increase the risk where the constraint is active
6. End loop
Iterative Risk Allocation Algorithm

Algorithm IRA

1. Initialize with arbitrary risk allocation
2. Loop
3. Compute the best available path given the current risk allocation
4. Decrease the risk where the constraint is inactive
5. Increase the risk where the constraint is active
6. End loop

No gap = Constraint is \textit{active}

Gap = constraint is \textit{inactive}

Goal

Start

Safety margin

Best available path given the safety margin
Iterative Risk Allocation Algorithm

Algorithm IRA

1. Initialize with arbitrary risk allocation
2. Loop
3. Compute the best available path given the current risk allocation
4. Decrease the risk where the constraint is inactive
5. Increase the risk where the constraint is active
6. End loop
Iterative Risk Allocation Algorithm

Algorithm IRA

1. Initialize with arbitrary risk allocation
2. Loop
3. Compute the best available path given the current risk allocation
4. Decrease the risk where the constraint is inactive
5. Increase the risk where the constraint is active
6. End loop
Iterative Risk Allocation Algorithm

Algorithm IRA

1. Initialize with arbitrary risk allocation
2. Loop
   3. Compute the best available path given the current risk allocation
   4. Decrease the risk where the constraint is inactive
   5. Increase the risk where the constraint is active
6. End loop
Algorithm IRA

1. Initialize with arbitrary risk allocation
2. Loop
3. Compute the best available path given the current risk allocation
4. Decrease the risk where the constraint is inactive
5. Increase the risk where the constraint is active
6. End loop
Iterative Risk Allocation Algorithm

**Algorithm IRA**

1. Initialize with arbitrary risk allocation
2. Loop
3. Compute the best available path given the current risk allocation
4. Decrease the risk where the constraint is inactive
5. Increase the risk where the constraint is active
6. End loop
Iterative Risk Allocation Algorithm

Algorithm IRA

1. Initialize with arbitrary risk allocation
2. Loop
   3. Compute the best available path given the current risk allocation
   4. Decrease the risk where the constraint is inactive
   5. Increase the risk where the constraint is active
   6. End loop
pSulu RH

VERY roughly speaking...

\[ \text{pSulu RH} = \text{Iterative Risk Allocation} + \text{Receding horizon control} \]
Receding Horizon Control

- Patchwork.

First $N$ steps

Goal

Start
Receding Horizon Control

- Patchwork.
Receding Horizon Control

- Patchwork.

![Diagram showing the concept of Receding Horizon Control with labeled steps: Start, First $N$ steps, Next $N$ steps, and Goal.]
Risk Budgeting

\[ \Delta = 1\% \]
Risk Budgeting

Planning horizon
Execution horizon

Start

$t = 0$

$t = 1$

$t = 2$

$t = 3$

$t = 4$

Goal

Obstacle

Risk budget

$\Delta A$

0

0.1%

0.5%

1.0%
Risk Budgeting

Planning horizon
Execution horizon

Goal
Obstacle

Risk budget

$t = 0$
$t = 1$
$t = 2$
$t = 3$
$t = 4$

0.1%
0.5%
1.0%

Start

$	riangle A$
Risk Budgeting

Start

Obstacle

Goal

0.1%

0.3%

0.4%

0.5%

1.0%

Risk budget

\[ \Delta A \]

MIt
Risk Budgeting

Goal

Start

Obstacle

t = 0
t = 1
t = 2
t = 3
t = 4
t = 5
t = 6

0.1%
0.4%
0.3%

Risk budget

Δ A

1.0%
0.5%
0.5%
0
Goal
Start
Obstacle

Risk Budgeting

0.1%
0.4%
0.3%
0.1%

Risk budget

0.5%
1.0%
0
Risk Budgeting

Goal

Start

Obstacle

\[ t = 0 \]

\[ t = 1 \]

\[ t = 2 \]

\[ t = 3 \]

\[ t = 4 \]

\[ t = 5 \]

\[ t = 6 \]

Risk budget

0.1%

0.4%

0.3%

0.1%

0.5%

1.0%
Result: p-Sulu

- Risk-performance trade-off
  - More risk ↔ shorter path
  - Less risk ↔ longer path
p-Sulu Application to Space Rendezvous

HTV unmanned resupply vehicle

Challenges:
- Risk of collision
- Complicated rendezvous procedure
- Unintuitive dynamics (follows Clohessy-Wiltshire eq.)

AI: Approach Initiation
RI: R-bar Initiation
HP: Hold Point
PP: Parking Point
CB: Capture Box

ISS Orbit
AI Point
x
y

RI
HP
PP
CB

CB: Capture Box

AT LUMB
-5000 m
HTV rendezvous planning problem

HTV unmanned resupply vehicle

Remain in [RI]
Remain in [YA]
Remain in [PP]
End in [RP]

Chance constraints:

\[ \Delta_1 = 0.5\% \]
\[ \Delta_2 = 0.5\% \]
AI: Approach Initiation, RI: R-bar Initiation, YA: Yaw-around

(a) Earth
(b) RI point $t = 2280$  YA point $t = 3120$
(c) Parking point $t = 3960$  Rendezvous point $t = 4800$

Massachusetts Institute of Technology

Direction of motion of ISS
### HTV rendezvous planning : Result

<table>
<thead>
<tr>
<th></th>
<th>Algorithm</th>
<th>Sulu</th>
<th>p-Sulu</th>
<th>Nominal</th>
</tr>
</thead>
<tbody>
<tr>
<td>$c_1$ (Navigation)</td>
<td>Risk bound $\Delta_1$</td>
<td>0.005</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Probability of failure $P_{\text{fail},1}$</td>
<td>0.92</td>
<td>0.0024</td>
<td>$&lt; 10^{-6}$</td>
</tr>
<tr>
<td>$c_2$ (Goals)</td>
<td>Risk bound $\Delta_2$</td>
<td>0.005</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Probability of failure $P_{\text{fail},2}$</td>
<td>1.0</td>
<td>0.0029</td>
<td>$&lt; 10^{-6}$</td>
</tr>
<tr>
<td>Cost function value (Delta V) $J^*$ (m/s)</td>
<td>7.30</td>
<td>7.32</td>
<td>8.73</td>
<td></td>
</tr>
<tr>
<td>Computation time (s)</td>
<td>3.9</td>
<td>11.4</td>
<td>0.09</td>
<td></td>
</tr>
</tbody>
</table>

11.9 kg saving of fuel, compared to the nominal plan