Massachusetts Institute of Technology

"Progress Towards Task-Level Collaboration Between Astronauts and Their Robotic Assistants"

Robert Effinger, Andreas Hofmann, Prof. Brian Williams

Model-Based Embedded and Robotic Systems Group
Massachusetts Institute of Technology

ISAIRAS 2005

Model-based Embedded & Robotic Systems

Task Level Collaboration with Robotic Assistants

ATRV Rover Testbed — Humanoid Simulators

These Robotic Assistants must be able to:

1.) Interpret task level commands

2.) Execute the tasks safely and reliably, even under disturbances and execution uncertainties

1.) Interpreting task level commands

Reactive Model-Based Programming Language
(Williams et.al., ISAIRAS 01)

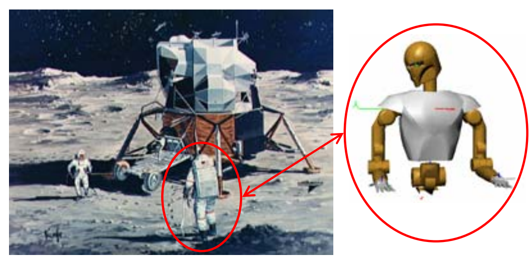
```
RMPL := c | act | TPN1 |
    TPN1 [ lb , ub ] |
    parallel ( TPN1 , TPN2 , ... ) |
    sequence ( TPN1 , TPN2 , ... ) |
    choose ( TPN1 , TPN2 , ... ) |
    if (c) thennext ( TPN1 ) |
    do ( TPN1 ) maintaining (c)
```

Temporal Plan Network
(Kim, Williams, Abrahmson IJCAI 01)

A [lb , ub]	A [lb , ub]
c [lb , ub]	[lb , ub] Tell (c)
sequence (TPN1 , TPN2 , ...)	TPN1 TPN2 ...
parallel (TPN1 , TPN2 , ...)	TPN1 TPN2 ...
choose (TPN1 , TPN2 , ...)	TPN1 TPN2 ...
if (c) thennext (TPN1)	Ask (c) TPN1
do (TPN1) maintaining (c)	TPN1 Ask (c)

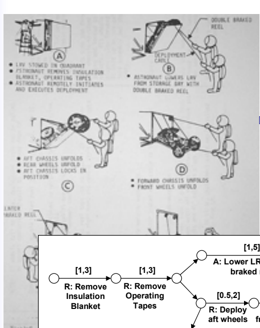
1.) Interpreting task level commands

A Motivating Example: Lunar Roving Vehicle Deployment



(*picture courtesy of NASA)

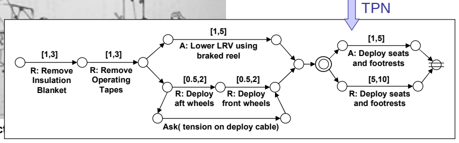
Apollo LRV Deployment Sequence



RMPL

```
LRV-deployment-sequence() [5,20] = {
 sequence(
   R: Remove insulation blanket [1,3],
   R: Remove operating tapes [1,3]
   parallel(
     A: Lower LRV w/braked reel [1,5],
     do (
       sequence(
         R: deploy aft wheels [0.5,2],
         R: deploy front wheels [0.5,2]
       )
     )maintaining (tension on cable)
   )
   choose(
     A: deploy seats & footrests [1,5],
     R: deploy seats & footrests [5,10]
   )
 )
} [5,20]
```

TPN

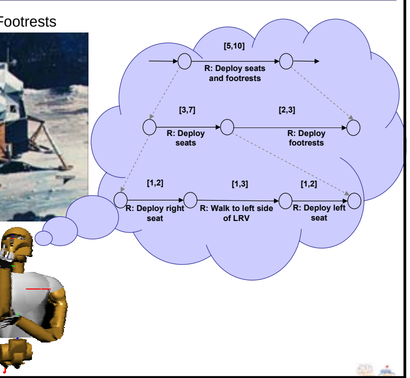


(*pic

Abstract Task Decomposition

Deploy Seats and Footrests

(*picture courtesy of NASA)



Task Level Collaboration with Robotic Assistants

ATRV Rover Testbed — Humanoid Simulators

These Robotic Assistants must be able to:

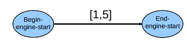
1.) Interpret task level commands

2.) Execute the tasks safely and reliably, even under disturbances and execution uncertainties

Flexible Execution Times

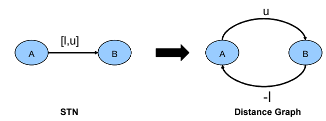
(Dechter, Meiri, Pearl 91)

Simple Temporal Network (STN):



$$T_j - T_i \in [1,5]$$

Equivalent Distance Graph Representation:

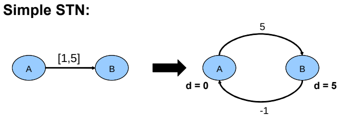


$$T_j - T_i \in [l,u] \quad \Rightarrow \quad T_j - T_i \le u \cap T_i - T_j \le -l$$

Flexible Execution Times

(Dechter, Meiri, Pearl 91)

A Simple STN:



Consistent !

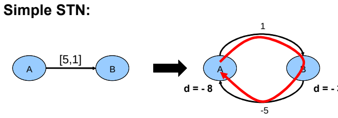
Determine STN consistency:

- **Calculate the Single Source Shortest Path (polynomial-time algorithm)**

Flexible Execution Times

(Dechter, Meiri, Pearl 91)

A Simple STN:



Inconsistent STN !

Determine STN consistency:

- **Calculate the Single Source Shortest Path (polynomial-time algorithm)**
- **A continually looping negative cycle indicates an inconsistency in STN**

Two methods to detect a continually looping negative cycle

1.) Check for any d-value to drop below –nC. (most space efficient)

2.) Keep an acyclic spanning tree of support, and terminate when a self-loop is formed. (Cesta, Oddi 96) (most time efficient)

Incremental Reasoning Algorithm

- Basic Idea:

  1.) Keep dependency information for each shortest-path value in the distance graph (Cesta, Oddi 96)

  2.) Use incremental update rules to localize necessary changes to the distance graph.
  - a.) 3 Update Rules to change a consistent distance graph.
  - b.) 3 Update Rules to repair an inconsistent distance graph.

- ITC's Novel Claims:

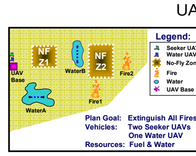
  1.) A conflict extraction mechanism to guide plan repair

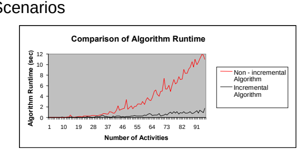
  2.) Allow multiple arc-changes

  3.) Can repair inconsistent distance graphs incrementally

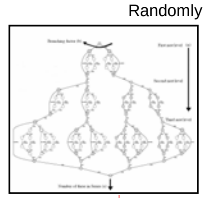
Performance Improvements

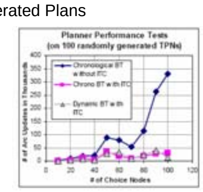
UAV Scenarios



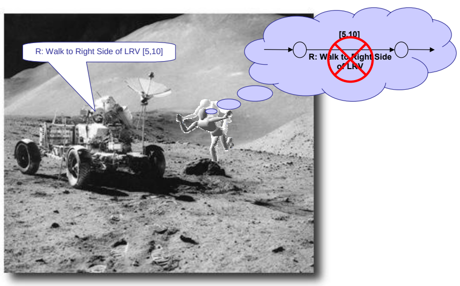


Randomly Generated Plans



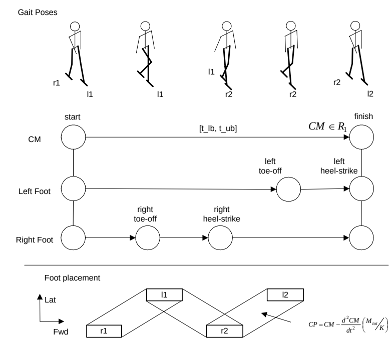


Robustness To Environmental Disturbances and Uncertainty – Andreas Hofmann, PhD Thesis



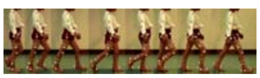
Qualitative Behavior Specification for Locomotion

- [Muybridge, 1955]
- Stop-action photographic study of human and animal motion
- Gaits depicted as sequences of distinct qualitative poses



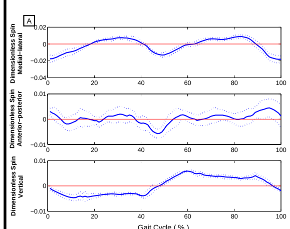
Flexible spatial and temporal constraints

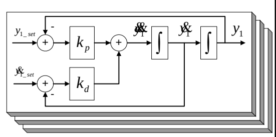
Nominal Walking



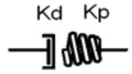
- Angular momentum tightly conserved during normal walking

- Allows for linearizing controllers that decouple state variables and makes them directly controllable
  - [Hofmann, et al; 2004]

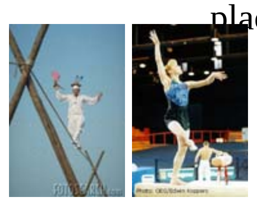




Kd Kp



Walking with constrained foot placement

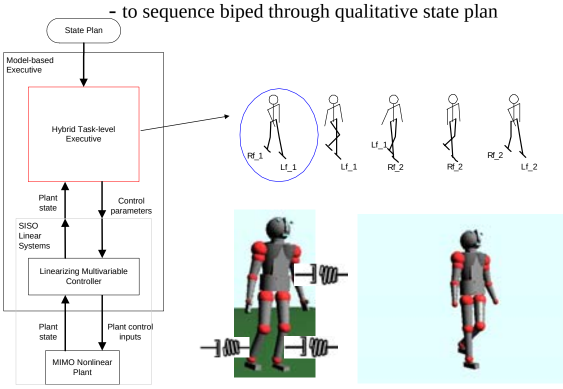


- Implemented in controller through Lagrangian relaxation
  - Orientation goals lower priority than balance goals

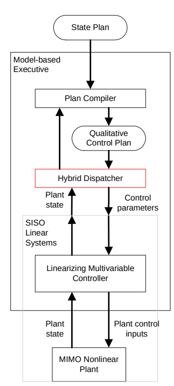
- When disturbed, sacrifice tight angular momentum conservation temporarily
  - Until balance restored

Hybrid executive coordinates controllers

- to sequence biped through qualitative state plan



Plan compilation for efficient execution



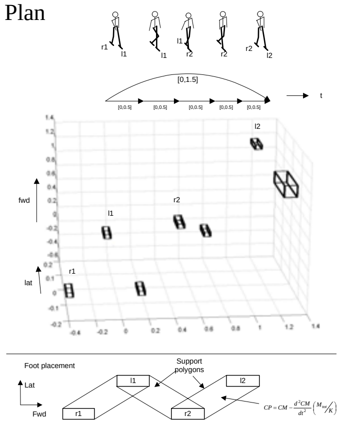
Compiler:

- Computes tubes of feasible control trajectories from Qualitative State Plan.

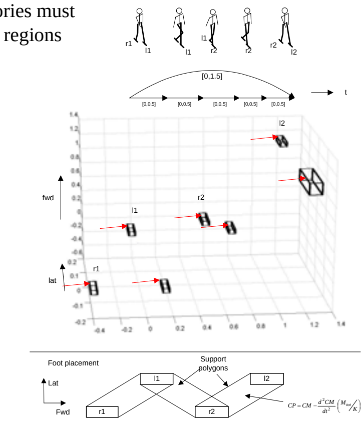
Dispatcher:

- dynamically searches for optimal control trajectories within tubes.
- Dispatcher "pulls springs" of each state variable.

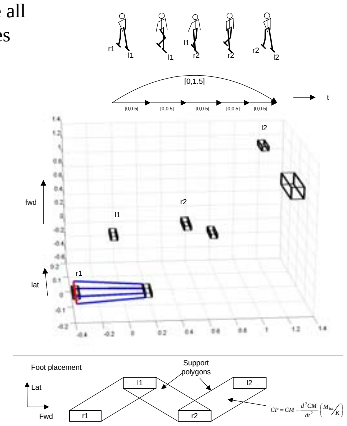
Qualitative State Plan
Goal Regions



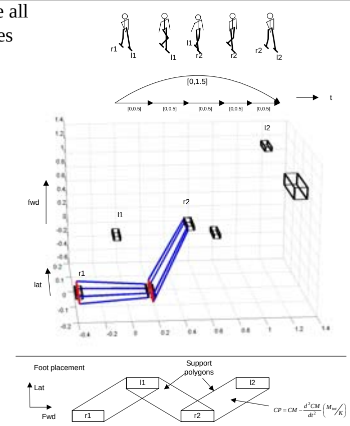
Feasible trajectories must go through goal regions



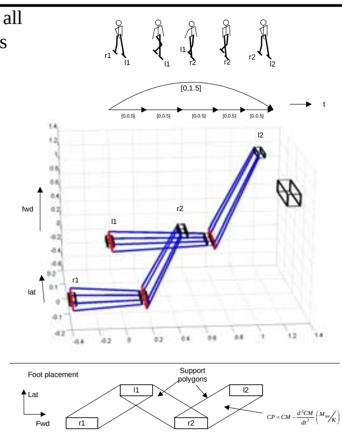
Flow tubes denote all feasible trajectories



Flow tubes denote all feasible trajectories

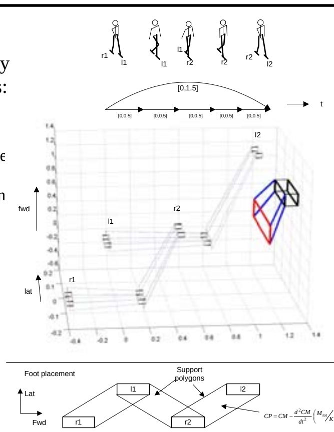


Flow tubes denote all feasible trajectories

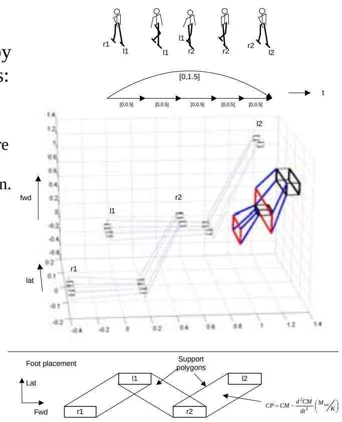


Center of Mass CM tube constrained by foot position tubes:

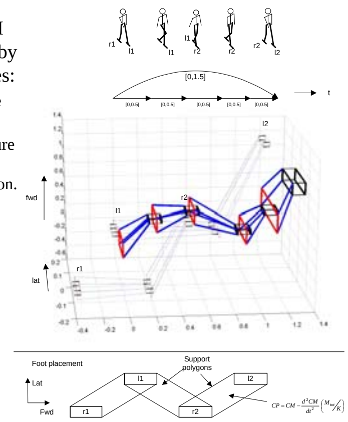
- Foot positions define support polygon..
- Center of foot Pressure CP constrained to be inside support polygon
- CM coupled to CP.



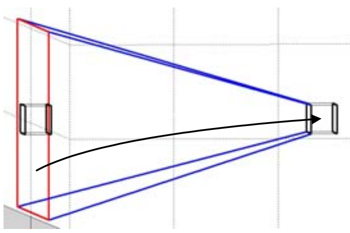
Center of Mass CM tube constrained by foot position tubes:
- Foot positions define support polygon..
- Center of foot Pressure CP constrained to be inside support polygon.
- CM coupled to CP.



Center of Mass CM tube constrained by foot position tubes:
- Foot positions define support polygon..
- Center of foot Pressure CP constrained to be inside support polygon.
- CM coupled to CP.

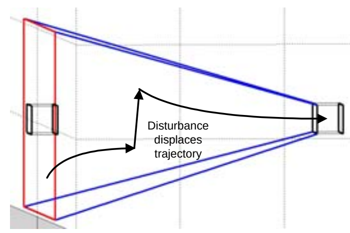


Successful execution of activity



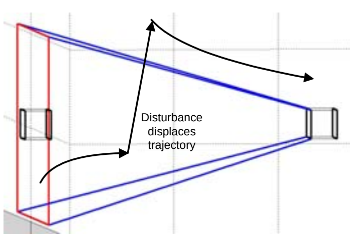
- Execution successful if trajectory stays in state tube.
- If trajectory begins in initial region, and no disturbances, will stay in tube.
- If trajectories of each activity stay in tube, plan executes successfully.

Disturbance displaces trajectory in state space



- If disturbance not too large, displacement stays in tube.
- Activity still executes successfully.

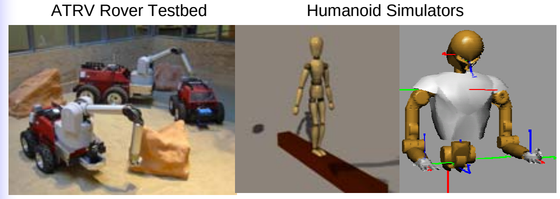
Disturbance displaces trajectory in state space



- If disturbance too large, trajectory pushed outside tube.
- Goal region not achievable at the required time.
- Plan failure detected immediately leaving more room for recovery.

Conclusions

ATRV Rover Testbed Humanoid Simulators



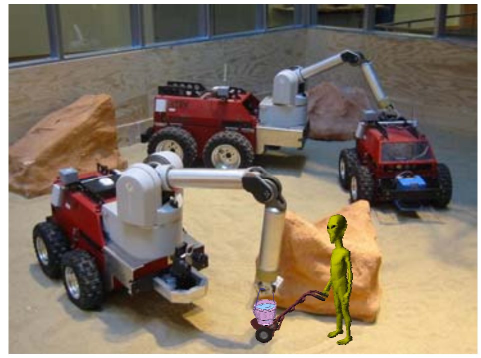
These Robotic Assistants must be able to:

1.) Interpret task level commands
with the Reactive Model-based Programming Language

2.) Execute the tasks safely and reliably, even under disturbances and execution uncertainties
with Temporal Plan Networks, Qualitative State Plans, and Flow Tubes

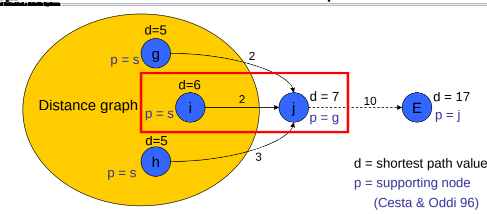
Any Questions?

Any Questions?



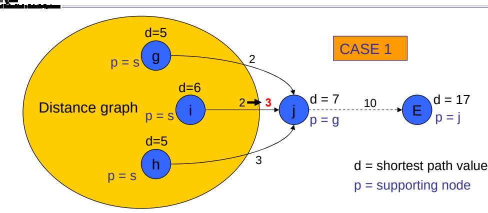
Extra Slides

3 Update Rules to Change a Consistent Distance Graph



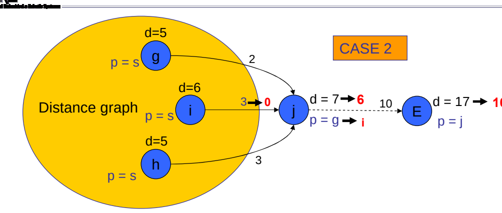
- Keep track of the support for each shortest path value
- Given a consistent STN, changing Arc(i,j)'s cost can have three possible effects on the shortest path.
    1. Arc(i,j) change does not affect the shortest path to node j.
    2. Arc(i,j) change improves the shortest path to node j.
    3. Arc(i,j) change invalidates the shortest path to node j.

1.) Arc(i,j) change does not affect shortest path



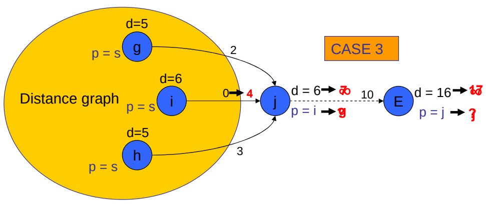
- The cost from node *i* to node *j* increases from *2* to *3*.
- No changes are needed.

2.) Arc(i,j) change improves shortest path to *j*



- The cost from node *i* to node *j* decreases from 3 to 0.
- Propagate the improved shortest path.

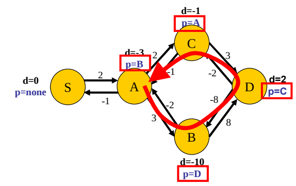
3.) Arc(i,j) change invalidates shortest path to *j*



- Increasing Arc(i,j) now invalidates node *j's* shortest path.
  - Reset node *j*
  - Recursively reset nodes dependent upon node j.
  - Insert node *j's* parents into the queue so that a new path to node *j* can be found for node *j* and all other invalidated nodes.

3 Update Rules to repair an Inconsistent distance graph

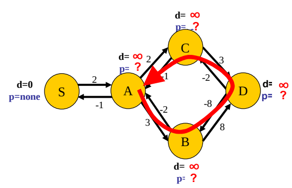
- ITC discovers an inconsistency (a negative cycle) by detecting cyclically dependent backpointers.



This must be a negative cycle cycle: (A,B,D,C,A)

3 Update Rules to repair an Inconsistent distance graph

- Now ITC must incrementally repair the inconsistency.

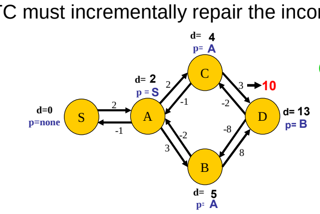


Negative cycle: (A,B,D,C,A)

- Three repair steps:
  1.) Reset all nodes in negative cycle.
  2.) Recursively reset all nodes that depend on the negative cycle nodes.
  3.) Put any parent of a reset node that was not also reset on the Q.

3 Update Rules to repair an Inconsistent distance graph

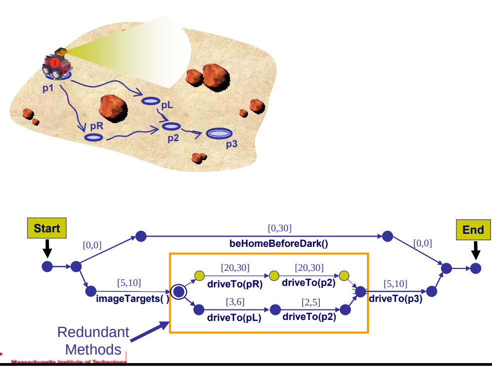
- Now ITC must incrementally repair the inconsistency.



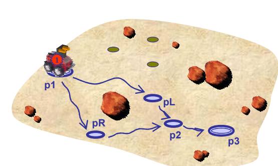
Consistent !

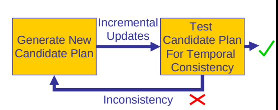
- Change arc cost CD to 10.
- Propagate the new shortest path values

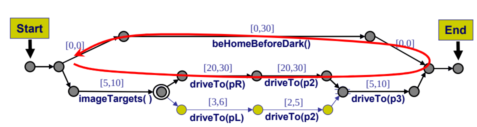
Temporal Plan Network (Kim, Williams, Abrahmson 01)



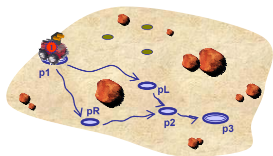
Conflict-Directed Plan Repair

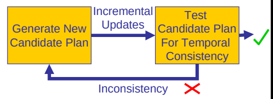


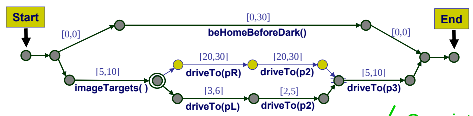




Conflict-Directed Plan Repair







√ Consistent !

Conclusions

- ITC is an incremental shortest path algorithm that can repair distance graphs incrementally as the plan changes
- ITC's Novel Claims:
  1.) A conflict extraction mechanism
  2.) Allow multiple arc-changes at once
  3.) Can incrementally repair inconsistent distance graphs
- Shows an order of magnitude improvement over non - incremental planning
- Applicable to any plan representation that uses disjunctions of simple temporal constraints.