From Task to Motion Planning

David E. Smith
NASA Ames Research Center
1997-2000: Marsokhod
2009-: A Different Kind of Rover
Some Rover Peculiarities

- Hazardous environments
  - Slow rad-hardened processors (200 MHz)
  - Low power (125 Watts)
  - Limited memory (256 MB)
  - Limited storage (2 GB)

- Unstructured rough terrain
  - Navigation/localization difficult

- Limited autonomy
  - Local obstacle avoidance
  - Opportunistic pictures
Some Rover Peculiarities

- Hazardous environments
  - Slow rad-hardened processors (200 MHz)
  - Low power (125 Watts)
  - Limited memory (256 MB)
  - Limited storage (2 GB)

- Unstructured rough terrain
  - Navigation/localization difficult

- Limited autonomy
  - Local obstacle avoidance
  - Opportunistic pictures
The Planning Problem

• Temporal
  - Action durations
  - Concurrency

• Time constraints
  - Communication windows
  - Illumination of targets
  - Temperature

• Uncertainty
  - Terrain & tracking
  - Duration of actions
  - Energy usage
  - Storage available

• Oversubscription
  - Many conflicting goals
  - Goal dependence
ATHLETE

- 6 legs, 36 degrees of freedom
- Feet are wheels (walk and roll)
- Tool takeoff on each wheel
ATHLETE

- 6 stereo camera pairs outward
- 3 stereo camera pairs inward
- 1 stereo pair on each foot
- 2.75m chassis
- 850 kg
ATHLETE in action
Current Operation

- Remotely operated
- Rolling: ok
- Walking: slow
Walking

- Raise foot 10 cm
- Raise foot 40 cm
- Rotate hip 60 degrees
- Pitch knee 40 degrees
- Pitch ankle -40 degrees
- Rotate hip 10 degrees
- Lower foot 40 cm
- Lower foot 10 cm
Simple Planning Problem

• Given:
  – simple goal point
  – terrain map with varying resolution
    • detailed ≤ 5 meters
    • satellite > 5 meters

• Find:
  – command sequence
  – prefer rolling to stepping
Discretization

Cartesian Space:
Discretization

Cartesian Space:
Discretization

Cartesian Space: ????

- Collisions of the entire leg
- Not every Cartesian path can be followed
- Not a 1-1 mapping
Configuration Space

Point in 6D (joint angles)

Legal path in joint space

Points in 6D

Cartesian Space

Start

Goal

Thursday, June 6, 2013
The specific goal of this paper is to evaluate four multi-step walking and do not include them here. We can represent the location of the foot as either a Point in 6D (joint angles) or a Point in 6D. The legal path in joint space must respect physical constraints on the robot and no collisions with self or environment. Points in 6D must also be free of collisions and the leg must be collision free. Ultimately, the solution path must be truly free of the ground due to the way the chassis will sag. Given this data, we compute start and goal configurations. This includes the location and orientation of the chassis remain fixed and therefore that the start and end configurations our second is a standard path through configuration space represents a sequence of edges and the tire will expand as the leg is lifted. While the Raise/Lower commands are done in task space, we assume that this represents a valid and stable edge. Three of our four algorithms search through a discretization of configuration space. While the Straight line edge between start and goal is one-to-one, it could correspond to many different configurations. Our first algorithm only tries the straight line between start and goal for a given leg. Our second approach is a Single-query Bi-directional Search. As problem input, we assume the start location, the goal location, and the current position. This includes the location and orientation of the leg. We assume that this represents a valid and stable configuration. Finally, we have a function TO-CSPACE that determines whether the straight line in configuration space is free of collisions. The solution path must be collision-free. Ultimately, the solution path must be truly free of the ground due to the way the chassis will sag.
Planning in Configuration Space

- **Discretize**
- **A**
- **Stochastic Local Search (SBL)**

Point in 6D (joint angles)

Legal path in joint space

Points in 6D
Planning in Configuration Space

Point in 6D (joint angles)

Probabilistic Methods

Legal path in joint space

Points in 6D
Planning in Configuration Space

![Diagram of configuration space with points and paths]

- **Point in 6D (joint angles)**
- **Legal path in joint space**
- **Points in 6D**

**Probabilistic Methods**

- **PRMs**
- **A***

**A six-tuple in configuration space**

- For a given start and end configurations, our second approach is a standard randomized motion planning algorithm, and our third is an Az search through a discretization of configuration space.

- The goal for each algorithm is to produce a path through configuration space such that each path represents a sequence of allowable moves connecting the start and end configurations.

- Our first algorithm only tries the straight line between the start and end configurations. Our second approach is a Single-query Bi-directional Probabilistic Motion Planner (SMPL) that is free of collisions. If the straight line between two configurations is collision free, the algorithm successively refines the path using the Lazy collision checking technique. If not, the algorithm fails.

- Our different approaches to generate the path make different assumptions about how the robot can move. The specific goal of this paper is to evaluate the effectiveness of these approaches in generating a path through configuration space for the robot. We consider such motions part of the robot's configurational degrees of freedom, and we can ignore the configuration of the other five legs.

- The leg bearing leg must be raised by about this much before it is allowed to move. We assume that the position and orientation of the chassis remain fixed, and therefore, that the motion is confined to the six joint angles for each leg. The leg orientations remain fixed, and therefore, that the motion is confined to the six joint angles for each leg.

- Notice that in the example command sequence, while the Raise/Lower commands are done in task space, the sequence of commands to move an ATHLETE foot from one configuration to another represents a sequence of motions of the robot's legs as the robot walks.

- The start and end configurations are not collision free. Ultimately, the solution path is acquired with ATHLETE's onboard terrain camera. For our experiments, it is automatically generated.

- The Terrainsketch simplification means that we are only concerned with the six joint angles of each leg and the relative bearing leg orientations of each leg. We include these 65 cm buffers because a weight of 1 kg is added to the robot in reality, but it is acquired with ATHLETE's onboard terrain camera.

- The Terrainsketch suggests future directions and we conclude in Section 7.

- In Section 7, we explain the four algorithms used.
Planning in Configuration Space

Probabilistic Methods

Point in 6D (joint angles)

Legal path in joint space

Points in 6D

Grow trees (RRT)

2.3. SBL

Our second approach is a Single-query Bi-directional

SMPL

Our baseline algorithm

Our fourth and final approach is Az search in

task space

Three of our four algorithms search

in Algorithm 63

2.2. Straight line approach

The goal for each of our algorithms is to produce a

path through configuration space represents a sequence of

commands to move an ATHLETE foot from

while the Raise4Lower commands are done in task space

the three-dimensional Euclidean space in which the robot

randomized motion planning algorithm and our third is an

from one configuration to another

path through configuration space represents a sequence of

walking

The specific goal of this paper is to evaluate four

not collide with itself, other parts of the robot, or the terrain

that determines whether the straight line in configuration

spaces via the forward or inverse kinematics of the leg

As problem input, we assume

We can represent the location of the foot as either

in joint space

...
Problems with Prob. Methods

- Optimality
- Awkward paths
- Narrow channels
- Non-repeatability
- Active compliance
Path Smoothing
Path Smoothing
Path Smoothing
Path Smoothing
Problems with Prob. Methods

- Optimality
  - enough points, A*, smoothing

- Awkward paths
  - smoothing

- Narrow channels
  - smarter points

- Non-repeatability
  - roadmap retention

- Active compliance
  - sequencing
Simple Planning Problem

• Given:
  – simple goal point
  – terrain map with varying resolution
    • detailed $\leq$ 5 meters
    • satellite $>$ 5 meters

• Find:
  – command sequence
  – prefer rolling to stepping
Dumb Idea #1

Joint space planning for entire robot
- (all 6 legs + shifting + rolling)
Almost as Dumb Idea #2

Joint space planning for all six legs

- 16 minutes on flat terrain
- 27 minutes on rough terrain
Decomposition

- Sequence of locations
- Rolling, Rotating, Shifting, Stepping
- Footfalls
- Joint planning for Steps

- Computational
- Data quality degrades quickly over distance
- Uncertainty regarding future configurations
Decomposing the problem

Route Planner
  - Goals
  - Viability

Chassis Planner
  - Goals
  - Viability

Move Planner
  - Goals
  - Viability

Leg Planner
  - Commands
  - State Update
Route Planner

• Given:
  – simple goal point
  – terrain map at varying resolution

• Find: route

• Simplifications:
  – robot is single point
  – terrain roughness as cost
Route Planning Approach

- Regular tessellation
- For each tile
  - steepness = max - min elevation
  - steepness < clearance
  - roughness = std-deviation from mode
  - cost = roughness * steepness

- Overstuffed tiles
Route Planning Search
Route Planning Search

- A*
  - D*-Lite
  - distance heuristics
    - n * green
color-cost = [#g, #y, #o, #r, #b]
Route Planning Search

- **A***
  - D*-Lite
  - distance heuristics
    - n * green
    - m * color-cost + n-m * green

\[
\text{color-cost} = [\#g, \#y, \#o, \#r, \#b]
\]
Route Planning Search
Route Planning Search

- A*
  - D*-Lite
  - distance heuristics
    - n * green
    - m * color-cost + n-m * green
    - n * color-cost
Chassis Planner

• **Given:**
  – goal direction, horizon, detailed terrain map

• **Find:**
  – sequence of translations and rotations
  – minimize stepping

• **Simplification:**
  – fixed leg pose
Chassis Planner

Rocks

Rocks
Chassis Planner Approach

• Fine tessellation of horizon
• For each tile
  – steepness = max - min elevation
  – steepness < clearance (within entire chassis)
  – roughness = std-deviation from mode
  – cost = roughness*steepness
• Overstuffed tiles
Chassis Planner Approach

• For successive chassis positions, cost is:
  – sum over leg paths of tile transition costs

• Additional penalties when
  – adjacent legs have significant elevation change at same time
Move Planner

- **Given:** fixed path for chassis
- **Find:** sequence of moves
  - Roll
  - Shift chassis
  - Step
- **Simplification:** delay collision checking

- drive 090,1m
- rotate -20
- roll-wheel 2, 20cm
- raise-leg 1
- drive 070,1m
- lower-leg 1
- step-leg 3, loc
- ...
Finding the best move

• Using depth-first search
  1. Roll if possible in the direction dictated by the chassis plan
  2. If lifting a leg will allow further rolling, prefer it
  3. If rotation will allow further rolling, prefer it
  4. For each leg and the chassis:
     • compute the max progress that the leg/chassis can be advanced in the direction of the chassis plan
     • order the leg/chassis moves according to progress along the chassis plan
Steps Considered

Reachable and stable regions are computed quickly by the Configuration Space routines.

R = Reachable positions
S = Stable positions
D = Desired positions
Move Planner
Move Planner
Move Planner
Move Planner
Move Planner
Move Planner

![Diagram showing a software interface for planning and scheduling with a terrain section and iteration labels.]

Thursday, June 6, 2013
Move Planner
Leg Planner

• Given: specific move
• Find: path in joint space
• No collisions
• Respect angle and torque limits
• Simplification: done in isolation
Stepping

- **Multi-step walking**
- **Chassis**
- **Goal**
- **Configuration of other five legs**
- **Future directions**
- **Section**

- **Rotate command**
- **Configuration space**
- **Euclidean space**
- **Randomized motion planning algorithm**
- **Third approach**
- **Path through configuration space**
- **Sequence of low-level commands**
- **Location**
- **Kinematics**
- **Algorithm**
- **Four algorithms**
- **Experiments**
- **Sections**

- **AThlete's joints**
- **Buffers**
- **Goal configuration**
- **Start and end configurations**
- **In addition**
- **Functions**
- **CM SPACE**
- **COLLISION-FREE**
- **Input**
- **Lazy collision checking**
- **Path planner**

- **Preliminaries**
- **Algorithms**
- **Straight line approach**
- **Single-query Bi-directional**
- **Approach**
- **Bi-directional**
- **Algorithm**
- **Computation**
- **Models**
- **Computations**

- **Joints**
- **Legs**
- **Robot**
- **Terrain**
- **Implementation**
- **Generated**
Stepping

Raise 10 cm

obstacle

Thursday, June 6, 2013
Stepping

Legal path in joint space

Point in 6D (joint angles)

Points in 6D

obstacle

Point in 6D (joint angles)
Algorithm 1: SMPL

- SMPL: Try straight line
Algorithm 2: SBL

- SBL: Single-query Bi-directional planner with Lazy collision checking
  - Grow two trees, occasionally try connecting

![Diagram showing two trees and a connection between them.](image-url)
Algorithm 3: CFG

- CFG: A* search in discretized 6D
Algorithm 4: TSK

- TSK: A* search in discretized 3D
Algorithm 4: TSK

- TSK: A* search in discretized 3D
# Comparison of path planning

<table>
<thead>
<tr>
<th></th>
<th>SMPL</th>
<th>SBL</th>
<th>CFG</th>
<th>TSK</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Space</strong></td>
<td>6D</td>
<td>6D</td>
<td>6D</td>
<td>3D</td>
</tr>
<tr>
<td><strong>Speed</strong></td>
<td>Fast</td>
<td>Fast</td>
<td>Slow and variable</td>
<td>Fast</td>
</tr>
<tr>
<td><strong>Quality</strong></td>
<td>Terrible</td>
<td>Good but variable</td>
<td>Mediocre</td>
<td>Good</td>
</tr>
<tr>
<td><strong>Smoothing</strong></td>
<td>NA</td>
<td>Crucial</td>
<td>Helpful</td>
<td>Helpful</td>
</tr>
</tbody>
</table>
Decomposing the problem

Route Planner

Chassis Planner

Move Planner

Leg Planner

Goals

Viability

Goals

Viability

Goals

Viability

Commands

State Update

Thursday, June 6, 2013
Example

Path Planner

At | Travel | At | Travel | At | Travel |

...
Example

Path Planner

At | Travel | At | Travel | At | Travel | ...

Chassis Planner

At | Translate | At | Rotate | At | Translate | At | ...

Thursday, June 6, 2013
Example

<table>
<thead>
<tr>
<th>Chassis Planner</th>
</tr>
</thead>
<tbody>
<tr>
<td>At Translate</td>
</tr>
</tbody>
</table>

Thursday, June 6, 2013
Example

Chassis Planner

At | Translate | At | Rotate | At | Translate | At

Move Planner

At | At

Thursday, June 6, 2013
Example

Chassis Planner

At → Translate → At → Rotate → At → Translate → At → ...

Move Planner

At → Roll → At → Shift body → At → Step → At → ...

Thursday, June 6, 2013
Move Planner

At | Roll | At | Shift body | At | Step | At

...
Example

Move Planner

At | Roll | At | Shift body | At | Step | At | ... |

Leg Planner

At | Roll | At | Shift body | At | a | b | c | ... | At | ... |

Raise foot 10 cm
Raise foot 40 cm
Rotate hip 50 degrees
Lower foot 40 cm
Lower foot 10 cm
Problem 1: Planning

Move Planner

| At | Roll | At | Shift body | At | Step | At | ... |

Leg Planner

| At | Roll | At | Shift body | At | × | × | ... |
Problem 1: Planning

Move Planner

At | Roll | At | Shift body | At | Stop | At | ... |

Leg Planner

At | Roll | At | Shift body | At | ... |
Problem 2: Execution

Move Planner

At Roll At Shift body At Step At

Leg Planner

At Roll At Shift body At a b c At

End up in different place or configuration

Thursday, June 6, 2013
Levels

Typical

- Task Planner
  - Route Planner
  - Arm Planner

More levels than usual

- Point
- Hexagon
- Body pose
- Individual leg

- Route Planner
  - Chassis Planner
  - Move Planner
  - Leg Planner

- Goals
- Viability
- Commands
- State Update

Thursday, June 6, 2013
Architectural Questions

• Level boundaries
  – Chassis x Move
  – Sequential vs Interleaved
    • Move & Leg
Architectural Questions

- Assymmetry
  - Inoperative Joint
  - Tool usage

Combine
More stability checking

Route Planner

Chassis Planner

Move Planner

Leg Planner

Goals

Viability

Goals

Goals

Viability

Goals

Viability

Commands

State Update

Traversability Heuristics

Thursday, June 6, 2013
Architectural Questions

• Collision checking
  – Route planner
  – Chassis planner
    • none
    • check frame
  – Move planner
    • none
    • check frame & non-moving legs
  – Leg planner
    • wheels only
    • leg
    • everything

Dependent on terrain difficulty ?
• Horizon
  – Route planner
  – Chassis planner
    • visual horizon ~ 5 meters
  – Move Planner
    • 2-5 meters
  – Leg planner
    • a few moves

Dependent on terrain difficulty?
Architectural Questions

• How often to replan at levels
  – Route planner
    • terrain detail changes roughness
    • cost of Chassis plan is higher than predicted
  – Chassis planner
    • cost of move plan is higher than predicted
    • advancement by more than 2 meters
  – Move Planner
    • after each command
    Dependent on terrain difficulty?
Architectural Questions

- Level breakdown
  - More than usual
  - Boundaries?
    - Chassis x Move
    - Sequential vs Interleaved

Diagram:
- Point
- Hexagon
- Body pose
- Individual leg

Flowchart:
- Route Planner
  - Goals
  - Viability
- Chassis Planner
  - Goals
  - Viability
- Move Planner
  - Goals
  - Viability
- Leg Planner
  - Commands
  - State Update

Thursday, June 6, 2013
Planning Assumptions

- Temporal
  - Action durations
  - Concurrency

- Time constraints
  - Communication windows
  - Illumination of targets
  - Temperature

- Uncertainty
  - Terrain & tracking
  - Duration of actions
  - Energy usage
  - Storage available

- Oversubscription
  - Many conflicting goals
  - Goal dependence

Levels of planning
• Temporal
  Action durations
  Concurrency

• Time constraints
  Communication windows
  Illumination of targets
  Temperature

• Uncertainty
  Terrain & tracking
  Duration of actions
  Energy usage
  Storage available

• Oversubscription
  Many conflicting goals
  Goal dependence
Complicating the Planning Problem

• Given:
  – collection of goals with utilities
  – time & resource constraints
  – uncertain durations & resource usage

• Find:
  – command sequence
  – prefer rolling to stepping
Impact

• Route Planner
  – need oversubscription planner

goals have utility
constraints on time & resources
maximize utility subject to constraints on time & resources

choose which goals to satisfy
Impact

- Route Planner
  - need oversubscription planner

goals have utility
constraints on time & resources
maximize utility subject to constraints on time & resources

choose which goals to satisfy

Net-Benefit Planner
goals have utility
actions have costs
maximize utility of goals

not the same!
Impact

• Other Levels?
  - Uncertainty in time and resource usage
    • impacts time constraints
    • constantly simulate expectations
    • more replanning required

Route Planner
  Goals
  Viability

Chassis Planner
  Goals
  Viability

Move Planner
  Goals
  Viability

Leg Planner
  Commands
  State Update
Contingency Planning

- Uncertainty in continuous quantity
- Discretization usually not viable
- Uncertainty is cumulative
  - the condition needs to be predictive
  - if probability of completing this goal drops below $x$, do plan2 instead
Making it more Real

- Temporal
  - Action durations
  - Concurrency

- Time constraints
  - Communication windows
  - Illumination of targets
  - Temperature

- Uncertainty
  - Terrain & tracking
  - Duration of actions
  - Energy usage
  - Storage available

- Oversubscription
  - Many conflicting goals
  - Goal dependence
Navigation and localization difficult
  - beyond horizon - only gross features from satellite images
  - choose paths near trackable features
Route Planning Search
cost = steepness * roughness * navigation-cost
Take Home Messages

- Multiple levels of planning
  - 4 levels of path planning
  - 3T+++  

- Good abstraction is key
  - allows feedback from lower level failures
  - minimizes backtracking between layers

- Task planning interacts primarily with highest layer
  - more serious with time constraints and duration uncertainty

- Levels break down with tool usage or damage