Automated Behavior Generation for Unmanned Ground Vehicles Using Virtual Environments

Satyandra K. Gupta, Bob Kavetsky, Steve Lubard, Krishnanand Kaipa, Madan Dabbeeru, Patrick Fry, Josh Langsfeld, Atul Thakur, and Petr Svec

Sponsor: Office of Naval Research
Objectives

• Develop a computational framework for automated generation of behaviors and maneuvers using simulations for autonomous UGVs
  
  – Behaviors are automatically generated and verified through simulations
  
  – Symbolic representation facilitates human verification
UGV System Architecture

Diagram showing the architecture of a UGV system with layers including Task Planning, Behavior Layer, Trajectory Planning, and Feedback Control. Key components include Task Planner, Behavior Selection, Behavior Executor, Model-Predictive Nominal Trajectory Planner, Model-Predictive Trajectory Refinement Planner, Feedback Controller, Mission, Traversability Model, Behaviors, Task Sub Goals Queue, Task Sub Goals Dispatcher, Motion Goal, Traversability Map, Maneuvers, Motion Primitives, Planning Heuristics, Controller Parameters.
Manually generating these libraries takes a long time! Any change in vehicle or mission context requires major updates to these libraries!
Overview of Approach

Synthesis by Evolutionary Computing

Statistical Inductive Reasoning

Learning from Demonstration

 Automatically Generated Software Libraries

Behaviors

Maneuvers

Traversability Models

Planning Heuristics

Planning Software

Route Planner

Behavior Selector and Executer

Trajectory Planner

UGV Commands

Simulation in Virtual Environment
Behavior Synthesis

- Use generalized behaviors:
  - **Input:** task objective, task constraint, estimated state
  - **Output:** motion goals and context dependent traversability constraints

- **Requirements:**
  - Satisfy task constraints
  - Meet task objectives
  - Produce realizable motion goals
  - Fast execution speed

**Actions:**

- $\pi : S \rightarrow A$
- $\pi_i : S_i \rightarrow A_i$

1) Splines
2) Mathematical expressions in terms of standard functions
3) Decision trees

- Transition boundaries
- Actions at boundaries are blends of actions of neighboring cells
UGV Simulator

• Modular architecture allows physics engine, vehicle controller, and machine learning algorithms to be kept separate
  – Both Unreal engine and high-fidelity Vortex vehicle dynamics engine implemented
• Terrain modeled manually or imported from Digital Elevation Model files representing real-world locations
• Vehicles
  – Are modeled in CAD/3D programs and imported into simulator
  – Have dynamics models that can be tuned for realistic behavior
  – Can be controlled by a human, scripts, or machine-learning algorithms
  – Can have their movement recorded and played back
• Multiple types of sensors can be emulated
  – Vision, velocity, acceleration, wheel spin, rangefinders, etc. are all modeled
• Additional static and dynamic objects, each with their own physical properties, can be added to the environment and controlled for machine learning experiments
Generate N-point turn maneuver between two arbitrary states $x_i = [x \ y \ \theta \ v]^T$ and $x_G = [x \ y \ \theta \ v]^T$, that satisfies the dynamics of the vehicle $f_M(x,u,t)$ and minimizes the execution time.

Generated maneuver should reliably be executed on a rugged, highly constrained, terrain.
Approach for Generating N-point Turn Behavior

Offline Synthesis

- Maneuver Synthesis Problem
  - Construct Atomic Maneuver Synthesis Problem
    - Update Atomic Maneuver Model
      - Atomic Maneuver Model
  - Synthesize Atomic Maneuver
    - Atomic Maneuver
    - Update Cost Function Policy Model
      - Cost Function Policy Model
  - Cost Function Policy Synthesis Problem
    - Construct Cost Function Policy Synthesis Problem
      - Update Cost Function Policy Model
        - Cost Function Policy Model
  - Turn Around Behavior

Online Execution

- Turn Around Behavior Execution
  - Motion Goals
    - Atomic Maneuver Selection Based on Terrain Inclination
      - Selected Maneuver
        - Maneuver Adaptation Based on Elevation Map
          - Adapted Maneuver
            - Turning Maneuver Feedback Loop Execution
Atomic Maneuver Model Synthesis

- Profiles of atomic maneuvers synthesized over a 5D space \((\delta x, \delta y, \delta \theta, \delta \alpha, \delta \beta)\) of
  - position and orientation differences \(\delta x, \delta y, \delta \theta\) between \(x_I\) and \(x_T\), and
  - terrain inclination angles \(\delta \alpha\) and \(\delta \beta\) w.r.t to \(x\) and \(y\) axis, respectively.

- Minimize: \(F = \omega f_1 + (1 - \omega)f_2\)
  - \(f_1\) – difference in position and orientation between \(x_I\) and \(x_T\)
  - \(f_2\) – maneuver execution time

- GA optimization
  - Initial population seeded with profiles obtained by manually driving vehicle between \(x_I\) and \(x_T\)
Motion Goals Computation

- Space exploration and computation of motion goals based on $x_I$, $x_T$, kinematic constraints, and obstacles
  - Planning and forward simulation of minimum number of forward and backward maneuvers
- Developed cost function policy model in which each policy is represented as a decision tree
  - Inputs:
    - Distance to target pose
    - Geometric configuration of obstacles
    - Maximum allowable clearances around the current and target poses
  - Outputs:
    - Position error weight $\omega_P$
    - Orientation error weight $\omega_\Theta$
    - Terrain error weight $\omega_T$
    - Parameterized curve
- Maneuver optimization using:
  $$C = \omega_P C_P + \omega_\Theta C_\Theta + \omega_T C_T$$
  $$C_P = \frac{||[x_T,y_T]^T-[x_I,y_I]^T||}{D_{max}}$$
  $$C_\Theta = \frac{-\dot{\text{dot}}(u,v)}{2} + \frac{1}{2}$$
  $$C_T = \begin{cases} 1 & \text{if collision} \\ 0 & \text{otherwise} \end{cases}$$
Experimental Results: 5-Point Turn Behavior Execution on Inclined Platform

Developed augmented reality setup for evaluation of N-point turn maneuvers
Follow Behavior

• Human walking on rough terrain exhibiting various types of search behaviors
  – Frenetic, Normal, Relaxed/Tired

• UGV follows human within distance boundaries defined by $r_{\text{min}}$ and $r_{\text{max}}$, while minimizing energy expenditure
  – $r_{\text{min}}$ and $r_{\text{max}}$ reflect estimated terrain type

• UGV stops and waits if inside zero activity zone defined by parameter $\lambda$

$\lambda$ difficult to learn!
Follow Behavior (Cont.)

- Behavior represented as a policy $\pi: S \rightarrow G$ that maps UGV state to motion goal $g = [x, y, \theta]^T$
  - Defines under what conditions to move and stay at current location based on $\lambda$

- Behavior input state includes
  - UGV heading error $\theta$
  - Operating zone boundaries $r_{min}, r_{max}$ that depend on terrain type
  - Execution length of specific type of human behavior $t_{h,max}$
  - Maximum human speed $v_{h,max}$ that depends on terrain type
Behavior Representation

\[ \pi: S \rightarrow G \]

**Meta-Action Selector**
\[ \pi_S: S \rightarrow \Lambda \]
for Follow Behavior

**Zero-activity zone parameter** \( \lambda \)

**Action Planner**
\[ \pi_P: \Lambda \rightarrow G \]
for Follow Behavior

**UGV state** \( s \)

**Terrain type** \((r_{\text{min}}, r_{\text{max}})\)

**Behavior representation as a composite of a decision tree for state space partitioning and a set of \( \Lambda \) functions**

<table>
<thead>
<tr>
<th>Behavior type</th>
<th>( h_{\text{min}} )</th>
<th>( h_{\text{max}} )</th>
<th>( v_{\text{min}} )</th>
<th>( v_{\text{max}} )</th>
</tr>
</thead>
<tbody>
<tr>
<td>Frenetic</td>
<td>1.5</td>
<td>5</td>
<td>3</td>
<td>30</td>
</tr>
<tr>
<td>Normal</td>
<td>0.5</td>
<td>1.5</td>
<td>3</td>
<td>60</td>
</tr>
<tr>
<td>Relaxed</td>
<td>0</td>
<td>0.5</td>
<td>0</td>
<td>120</td>
</tr>
</tbody>
</table>
Follow Behavior Synthesis Approach

- Defined optimization problem to find the appropriate $\lambda$ for each $S_i$ with respect to distance constraint violations
- Utilized Genetic Algorithm to generate $\lambda$ values for each subspace $S_i$
- Utilized NeuroEvolution of Augment Topologies (NEAT) to evolve neural networks as internal representations of $\Lambda$ functions
  - Symbolic rules were extracted from neural networks
Results:
Performance of Follow Behavior over Partitioned State Space

- Policy performance defined as

\[ P = f(\pi) = \frac{l_{UGV} + N_{cv} l_{penalty}}{l_h} \]

- Two alternative approaches were considered:
  a) optimal, constant \( \Lambda \) was determined for each cell in state space
  b) function was synthesized for each cell

- Performance of behavior improves with the increase in the number of cells in both cases
  - Approach b) has continuous input that leads to better performance

<table>
<thead>
<tr>
<th>Constant ( \lambda ) for all cells</th>
<th>( \Lambda : S_i \rightarrow \lambda ) function</th>
</tr>
</thead>
<tbody>
<tr>
<td># of cells</td>
<td>( P_{\text{mean}} )</td>
</tr>
<tr>
<td>----------</td>
<td>----------</td>
</tr>
<tr>
<td>12</td>
<td>0.71</td>
</tr>
<tr>
<td>36</td>
<td>0.69</td>
</tr>
<tr>
<td>360</td>
<td>0.54</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Baseline</th>
</tr>
</thead>
<tbody>
<tr>
<td>( P_{\text{mean}} )</td>
</tr>
<tr>
<td>1.31</td>
</tr>
</tbody>
</table>
Results

- Execution of baseline follow
- Performance
  - $P_{\text{mean}} = 1.31$
  - $P_{\text{stddev}} = 0.17$

- Execution based on state space partitioning
  - Constant $\lambda$ values for individual cells
  - Synthesized $\Lambda: S_i \rightarrow \lambda$ functions

- 12 cells, $P_{\text{mean}} = 0.71$, $P_{\text{stddev}} = 0.12$
- 360 cells, $P_{\text{mean}} = 0.54$, $P_{\text{stddev}} = 0.17$
- 1 cell, $P_{\text{mean}} = 0.81$, $P_{\text{stddev}} = 0.18$
- 12 cells, $P_{\text{mean}} = 0.44$, $P_{\text{stddev}} = 0.14$
Summary

• We are developing a new approach for automated behavior synthesis for UGVs

• We have shown that useful behaviors can be synthesized using simulations
  – N-Point Turn behavior
  – Follow Target behavior

• The next steps would be
  – Demonstrate that the behavior generation system can be easily reconfigured to generate new behaviors
  – Demonstrate that the synthesized behaviors can be executed on a physical platform without deterioration in performance
  – Demonstrate that the useful complex behaviors can be synthesized automatically
  – Perform experiments on a more capable physical platform equipped with GPS, IMU, lidar, and sonar sensors
For more information please contact:
Satyandra K. Gupta (skgupta@umd.edu)