Machine-Learning & Recommender Systems for C2 of Autonomous Vehicles

Glennn Moy

on behalf of

Don Gossink, Glennn Moy, Darren Williams, Kate Noack
Josh Broadway, Jan Richter, Steve Wark

Planning and Logistics, Decision Sciences, DST Group, Australia
Human-Autonomy Teaming

- Overall project – Balances two components:

**Autonomy**
Lead: Don Gossink

**HAT - Autonomy Team:**
- Don Gossink
- Glennn Moy
- Darren Williams
- Katherine Noack
- Josh Broadway
- Jan Richter
- Steve Wark
Recommendations for Command & Control of Multiple UxVs

Maritime Task Force
IMPACT

- U.S. Project aimed at:
  - Developing Intelligent Multi-UxV Planner with Adaptive Collaborative/Control Technologies
  - **Key Concept:** High-level, goal-oriented plays
Recommender System & Play-Monitor:

- Goal:
  - Develop advanced Recommender System(s) to reduce the cognitive burden on operators through:
    - Recommendations, alerts and constraints.
    - Lower-Level Autonomy

"Human on the loop"
Top-Level Architecture:

- User Context Logger
- Recommender System
- Simulation
- Human Play
- Play-Based / Manual Control
- Recommender GUI
- Recommender
- IMPACT / Generic Sim
HAT Challenge

- Limited access to IMPACT or other multi-UxV control system
- Need to integrate recommendations into various (unknown) system components

**Solution:**

- Recommender *loosely coupled* to the underlying system/simulation.
- Recommender that can *learn* recommendations at a range of C2 levels.
- Recommender techniques that work:
  - When heuristics are not known
  - In *new contexts* (not previously prepared)
Inspiration & Approach

- Build something that can learn... (like we learn???)

<table>
<thead>
<tr>
<th>Experience</th>
<th>Self Learning</th>
<th>Learning from Others</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Trying</td>
<td>Watching</td>
</tr>
<tr>
<td>Logic, Rules &amp; Heuristics</td>
<td>Reasoning</td>
<td>Explaining</td>
</tr>
</tbody>
</table>

Eg: Humans try out new moves and seeing what works.
Eg: Humans think about what works & what others might do in response to my moves.
Eg: Humans watch a good player and learn to recognise good board positions.
Eg: Humans learn by having a new strategy explained.

(NB: about gathering and structuring data to learn...not how we learn)
## Techniques & Requirements:

<table>
<thead>
<tr>
<th>Experience</th>
<th>Self Learning</th>
<th>Learning from Others</th>
</tr>
</thead>
</table>
| **Trying:** | • **Techniques:** Reinforcement Learning  
• **Requirements:** A simple model that machine can control that it can undertake re-enforcement learning on. | **Watching:**  
• **Techniques:** Supervised Deep Learning  
• **Requirements:** **Data** from expert playing a simulation that it can watch.  
  • Human Expert or Heuristic player that mimics an expert |
| **Logic, Rules & Heuristics** | **Reasoning:**  
• **Techniques:** Monte-Carlo State Exploration/Search, Logic, Planning.  
• **Requirements:** Implement efficient heuristic/search algorithms for exploring large state-space. | **Explaining:**  
• **Techniques:** Heuristics, Agent (BDI), Algorithms, Math/Planning, Abductive logic  
• **Requirements:**  
  • A language for expressing heuristic rules.  
  • Logic for constraining solutions.
Architecture:

- **Simulation**: Human Play
- **IMPACT / Generic Sim**: Play-Based / Manual Control
- **User Context Logger**: Heuristics and Rule Agents
  - Rule Constrainer(s) (Explaining)
  - Heuristic Player(s) (Explaining)
  - Futures Explorer (Reasoning)
- **Deep Learning Agents**: Reinforcement Learner(s) (Trying)
  - Feature Learner(s) (Watching)
- **Recommender System**: Recommend (Prioritise & Combine Agent Recommendations)
- **Recommender GUI**
- **Choice Constrainer**
- **Explain**
Recommendation Hierarchies

- **Recommender Agents:**
  - Implemented Recursively:
    - Hierarchy of recommendation agents.
  - Key Concepts:
    - Decomposing / Triggered Recommender Agents - *Elastic* autonomy
    - *Executable vs Non-Executable* Recommendations

- **Simulation:**
  - Publishes its own capabilities for accepting/executing recommendations.
1. Simulation

- Initial UAV Control Simulations
  - Modular/Plug-and-play
    - Ultimately to be replaced by IMPACT
  - Fast
    - In a training mode for rapid learning
  - Machine or Human controllable
2. First Heuristic Recommenders:

- Initial low-level *Executable* Recommenders
- Heuristics & Play Implementations
  - (a) Search for detection
    - Air-Expanding Square at/on a Point
    - Air Sector Search
    - Air Inspect Point
  - (b) Track Detections
3. First Reinforcement Recommender

- Initial Reinforcement Recommender
  - Deep Q Reinforcement Learning:
    - Combines reinforcement learning with deep neural network.
    - Originally used to play Atari Games – DeepMind/Google
  - Reinforcement Learning Challenges:
    - Credit-assignment problem
    - Explore-exploit dilemma
Deep Q Learning

Credit-Assignment Problem:

• Q(s, a) represents the maximum discounted future reward when we perform action a in state s (and get to state s’ with reward r.)
  - Q(s, a) = R(s, a) + γ \sum_{s'} T(s, a, s') \max_{a'} Q(s', a')
  - Q(s, a) = r + γ \max_{a'} Q(s', a')

• Iterative Updates (Training):
  - **Prediction**: Q(s, a)
  - **Target**: r + γ \max_{a'} Q(s', a')

\[ L = \frac{1}{2} \left[ r + γ \max_{a'} Q(s', a') - Q(s, a) \right]^2 \]

• Experience Replay:
  - Store all experiences <s, a, r, s'> in memory.
  - Train on random mini-batches

  - *Propagates rewards back in time.*
Deep Q Learning

Explore/Exploit Dilemma

• $\varepsilon$ – greedy exploration
  – With probability $\varepsilon$ choose a random action, otherwise go with highest Q value action.
  – $\varepsilon$ starts at 1.0 (always random) and slowly decreases (eg to 0.1 – mostly policy-based choices)
**Our Implementation**

- **Deep Q Network:**
  - Neural network with 2 hidden layers.
  - Theano/Keras implementation

- **Input State Data:** \((x, y, v_x, v_y, \text{sensor}_x, \text{sensor}_y)\)

- **Output:** Actions

---

**Initial Task:**

- Executable recommendation to track suspect vessel
- Feedback from the environment – Reward/Punishment

**Eg:** Move Location

- Up, Down, Left, Right

**Eg:** Change Velocity

- Rotate Left, Right
- Speed Up, Slow Down, Do Nothing

**Eg:** Switch Play

- Play 1, Play 6, Play 7

**Eg:** Correct direction = +ve Reward

- Incorrect direction = -ve Reward

**Eg:**

- Reward = -(# Successful Incursions)

- Reward = -(Distance from Goal)
Example 1:  

**Actions:** Up, Down, Left, Right  
(Constant Velocity, No Pause) 

**Reward:** -(Distance from Goal)
Example 2:  

**Actions:** Rotate Left, Right, Speed up, Slow Down, Do Nothing  

**Reward:** -(Distance from Goal)
Current/Ongoing R&D:

- **General:**
  - Implement Red “avoid detection” behaviour & blue agent vs red agent play
  - Learn/iterate to higher-level strategies / C2.

- **Deep Q R&D:**
  - Alternative Neural Architectures:
    - Impact on learning rate
    - Scalability with action-complexity
  - Multi-Agent Learning:
    - Single control agent vs multiple learning agents vs hybrid
  - Human-Guided Learning
    - Learning from human interactions as well as self-generated.

- **Other Recommender Agents:**
  - Agent Hierarchies
    - More complex (hierarchical) reward functions and interactions between recommender agents
  - Bayesian Inference for threat heat-maps
Questions