Multi-Agent Decentralized Planning for Adversarial Robotic Teams

James Edmondson
David Kyle
Jason Blum
Christopher Tomaszewski
Cormac O’Meadhra
Copyright 2016 Carnegie Mellon University

This material is based upon work funded and supported by the Department of Defense under Contract No. FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center.

Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the United States Department of Defense.

NO WARRANTY. THIS CARNEGIE MELLON UNIVERSITY AND SOFTWARE ENGINEERING INSTITUTE MATERIAL IS FURNISHED ON AN “AS-IS” BASIS. CARNEGIE MELLON UNIVERSITY MAKES NO WARRANTIES OF ANY KIND, EITHER EXPRESSED OR IMPLIED, AS TO ANY MATTER INCLUDING, BUT NOT LIMITED TO, WARRANTY OF FITNESS FOR PURPOSE OR MERCHANTABILITY, EXCLUSIVITY, OR RESULTS OBTAINED FROM USE OF THE MATERIAL. CARNEGIE MELLON UNIVERSITY DOES NOT MAKE ANY WARRANTY OF ANY KIND WITH RESPECT TO FREEDOM FROM PATENT, TRADEMARK, OR COPYRIGHT INFRINGEMENT.

[Distribution Statement A] This material has been approved for public release and unlimited distribution. Please see Copyright notice for non-US Government use and distribution.

This material may be reproduced in its entirety, without modification, and freely distributed in written or electronic form without requesting formal permission. Permission is required for any other use. Requests for permission should be directed to the Software Engineering Institute at permission@sei.cmu.edu.


DM-0004046
Challenges in Modern Unmanned Systems

- Current unmanned systems (UAS) are individually controlled by a handful of pilots and potentially dozens of analysts micromanaging every aspect of the device
- This control paradigm results in poor scalability and high training costs
- A centralized control station is also prone to failure, bottlenecks, and enemy attacks taking out all UAS managed by that station

High Level Takeaway

- Environments and missions change but AI tends to be static and preset
- Most unmanned systems use automated waypoints for missions

The current unmanned system practice of the DoD is one of micromanagement

Static AI is bad for dynamic adversaries
Micromanagement of Autonomous Systems is Pervasive

The State of Autonomy Middleware

- Robotic systems are built from the ground up
- Message queues are created between robotic components and controllers
- Autonomy developers must compose message-passing systems into something that supposedly works at a higher level
- No obvious way to check overall behaviors much less emergent behaviors
- ROS, UCS FACE (ARL), OMS (AFRL RCO) all force robotics developers to program around message queues

The arm, leg, head, etc. may be composed of dozens to hundreds of message queues
Our Autonomy Objectives

- Allow **one person** to **command an entire swarm** of UAS to do mission-level tasks

- **Focus on 1) scalability, 2) bringing simulated capabilities to reality,** and providing **3) predictable control** of UAS logic, threads, sensors, actuators and software components

- **Open source release** of middleware and software via BSD-style licenses at Sourceforge and GitHub (**GAMS/MADARA**)
Our Autonomy Process

- Users write an application in C++ or Java
  - Developers **read and write to knowledge** handled by the underlying middleware
  - **Platforms have standardized interfaces** that algorithms interact with
  - **No interaction with message queues** (handled under the hood)
- Users only have to **focus on the their algorithm or platform**
- Built-in translations between **simulation and real-world**
  - Pose system (Cartesian to GPS and vice-versa)
- **High consistency, predictability and QoS**
  - **Important for verification**

The result is rapid prototyping and verifiability of distributed autonomy in robotics (FY16 DART, SMC for Swarms)
Adversaries in MADPARTS

- Adversaries try to get line-of-sight on an important target (base, VIP vehicle, etc.)
- Adversaries move around a target, looking for an opening
- The goal of the new algorithms is to prevent line-of-sight on these targets
- Adversaries are modeled as agents
- Agents have self-interest and present information in the knowledge base like location to use in algorithm logic
- Essentially a persistent tracking system is assumed for tracking adversary position
Defensive Schemes

- We took some inspiration from American football and robot soccer

- **Zone defense:** Protector agents move to assigned zones between a vip and the enemy
  - Useful for holonomic robots like quadcopters

- **Onion defense:** Protector agents layer a defense between vip and enemy
  - Useful for non-holonomic robots like fixed-wing planes and boats that drift
Results: Simulations

- Algorithms were coded in C++ and made available via factory methods in GAMS (can be called remotely)
- The defensive algorithms were evaluated in VREP simulations
- The algorithms were evaluated in the FY16 SMC for Swarms Project (next talk) early in FY16
- With just 5 protectors, Line-of-sight was prevented at >99%

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Initial Disperse</th>
<th>Detect Range</th>
<th>Failure</th>
<th>Trials</th>
</tr>
</thead>
<tbody>
<tr>
<td>Scenario 1</td>
<td>Loose</td>
<td>Long</td>
<td>0.11%</td>
<td>265,896</td>
</tr>
<tr>
<td>Scenario 2</td>
<td>Loose</td>
<td>Short</td>
<td>0.35%</td>
<td>114,912</td>
</tr>
<tr>
<td>Scenario 3</td>
<td>Tight</td>
<td>Short</td>
<td>0.28%</td>
<td>114,504</td>
</tr>
<tr>
<td>Scenario 4</td>
<td>Tight</td>
<td>Long</td>
<td>0.00%</td>
<td>400,000+</td>
</tr>
</tbody>
</table>

http://coppeliarobotics.com/
Results: Real World Demonstration
Transition (ALW)

• PWP in place for AFRL Autonomy of the Loyal Wingman FY17-FY18
• Core software candidate for autonomous F-16 wingmen for a human pilot
• Algorithm creation for target defense and prosecution

https://en.wikipedia.org/wiki/General_Dynamics_F-16_Fighting_Falcon
Transition (NATO)

- Invitation to participate in NATO CMRE REP17-Atlantic exercise
  - REP17 is a joint exercise between Portuguese Navy, NATO CMRE, and the University of Porto
  - Current plan is for our autonomous boats to participate in the joint exercises

Boat images courtesy of Platypus LLC
Transition (Multi-Planetary Smart Tile)

- GAMS and MADARA are core software architecture for the Keck Institute for Space Studies’ Phase 1 Multi-Planetary Smart Tile
  - Hardware prototyped by GE GRC and Biovericom
  - Separate offers to launch into LEO by United Launch Alliance and NASA
  - Phase 1 is expected to perform simple autonomy experiments in low-earth orbit for up to 1 year

- Goal of project is to create a distributed, renewable power infrastructure for solar system that scales to tens of thousands of interacting robotic systems

Images courtesy of Anna Nesterova and Kelvin Ma
Conclusion

• Current autonomy practice suffers from:
  • Micromanagement of individual devices
  • Non-intuitive high-level behavior design and analysis
  • A distinct lack of verification tools

• Our middleware provides
  • Rapid prototyping capability for distributed autonomy
  • Full integration with DART and DEMETER (SMC for Swarms) for verification

• The MADPARTS defensive algorithms were successful
  • Prevented line-of-sight to target over 99.6% in all tested scenarios
  • Tested in unmanned surface vehicles in lakes near Pittsburgh
Contact Information
Name: Dr. James Edmondson
Title: Senior Member of the Technical Staff
Telephone: +1 412.268.8905
Email: jredmondson@sei.cmu.edu

Open Source Project Sites
MADARA: http://madara.sourceforge.net
GAMS: http://jredmondson.github.io/gams
DART: http://cps-sei.github.io/dart

SEI Contributors
James Edmondson
David S Kyle

CMU Contributors
Jason Blum
Christopher Tomaszewski
Cormac O’Meadhra

Platypus LLC
Paul Scerri paul@senseplatypus.com