Overview

- Project Goals and Objectives
- Learning Agent
- Problem Definition and Representations
- Environment Modeling and Simulation
- Results
- Flight Test
- Extensions
2011 – 2012 Research Team
Project Goals and Objectives
How It Is Done Now (1)

Payload Vehicle Operator

Air Vehicle Operator

Mission Command / Intelligence “Gatherers”
How It Is Done Now (2)
1. Identify a Preferred Concept for controlling a UAS and camera
   a. Keep a selected target visible in the camera field of view.
      • Frees a human supervisor to focus on selecting viable targets and analyzing the images received.

2. Develop a methodology for determining a combined UAS / image sensor control policy
   a. Guides UAS with a fixed-mounted image capturing device to track targets.
   b. Control policy shall track a pre-designated target in the field of view of the image capturing device.
   c. Type of image capturing device non-specific.

3. Track stationary and moving targets independent of road net / features / terrain data
Learning Agent
Inference Mechanism Candidates

- **Rules-Based Expert System**
  - Requires prior expert knowledge

- **Genetic Algorithms**
  - Not state-based domain
  - Does not save information learned between initial and final points

- **Fuzzy Logic**
  - Not state-based domain
  - Requires insights into behaviours that may not be possible

- **Artificial Neural Networks (ANN)**
  - Requires existence of good training actions data

- **Reinforcement Learning (RL)**
  - State-based domain
  - Requires reward information, but not training actions data
  - Model Free Method*
Reinforcement Learning 1

- Sequential decision making
  - Knowledge is based on experience and interaction with the environment, not on input-output data supplied by an external supervisor.

- Achieves a specific **goal** by learning from interactions with the environment.
  - Considers *state* information \( s \in S \)
  - Performs sequences of *actions* \( a \in A \), observing the consequences.
  - Attempts to maximize *rewards* \( (r) \) over time
    \[ R_i = r_{i+1} + \gamma r_{i+2} + \gamma^2 r_{i+3} + \ldots \sum_{i=0}^\infty \gamma^i r_{t+i} \]

  - These specify what is to be achieved, *not how to achieve it*.

- Constructs a *state value function* \( V(s) \) or *action-value function*, \( Q(s,a) \)
  - Memory is contained in the state value function,
    \[ V^*(s) = \max_\pi V^\pi(s) \]
  - or action-value function,
    \[ Q^*(s,a) = \max_\pi Q^\pi(s,a) \]

- Learns a control *policy*, \( \pi \), where \( \pi : S \rightarrow A \).
Reinforcement Learning 2

- Dynamic Programming (DP)
  - Historically DP assumed perfect knowledge of state transition and reward functions
  - Focused on minimal computational effort

- Q-Learning assumes no knowledge of state transition and reward functions
  - Can use real-world, not restricted to modeled world
    - Online (real world)
    - Offline (simulation)

- Bellman’s Equation
  \[ V^*(s) = \max_a E_r \{ r_{t+1} + \gamma V^*(s_{t+1}) \mid s_t = s, a_t = a \} \quad \forall (s \in S, a \in A(s)) \]

- RL Optimal Policy
  \[ V^\pi(s_t) = E_r \{ R_t \mid s_t = s \} = E_r \left\{ \sum_{k=0}^{\infty} \gamma^k r_{t+k+1} \mid s_t = s \right\} \]

  \[ V^*(s) = \max_\pi V^\pi(s) \]
Reinforcement Learning: Q-Learning

- Off-policy Method
  - Learned action-value function, $Q$, directly approximates the optimal solution, $Q^*$
    - Independent of the policy being followed

\[
Q(s_t, a_t) \leftarrow Q(s_t, a_t) + \alpha \left[ r_{t+1} + \gamma \max_a Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t) \right]
\]

- Policy determines state-action pairs

- **Mathematically proven convergence** to optimal policy

\[
Q(s_t, a_t) \rightarrow Q^*(s_t, a_t) \text{ as } N_{\text{visits}}(s_t, a_t) \rightarrow \infty
\]

but

\[
Q^\pi(s_t, a_t) \approx Q^*(s_t, a_t) \text{ if } N_{\text{visits}}(s_t, a_t) < \infty \text{ for } N_{\text{visits}} \text{ "large"}
\]

Watkins 1989
Problem Definition and Representation
Tracking as an RL Problem

- **States of System** $(s)$
  - **Target** x-position in image frame
  - **Target** y-position in image frame
  - UAS bank angle
  - $s = [X \quad Y \quad \phi]^T$
Tracking as an RL Problem

- **Goal of Learner** ($g$)
  - Move **target** to center of image
  - Once **target** reaches goal state, hold it there
  - Only **target** x-position and y-position considered for goal achievement
  - $g = [0 \quad 0 \quad \phi]^T$
Tracking as an RL Problem

- **Action Selection Rationale**
  - No control over target global position.
  - *Only way to track target in image frame is to steer the UAS itself.*
  - Under the current assumptions, bank angle is the UAS state that is controllable and has the greatest effect on target image position.
  - To lower number of state-action pairs to explore, change in commanded bank angle is used rather than simply commanded bank angle.

- **Actions** ($a$)
  - -2 degrees bank angle
  - +0 degrees bank angle (need to be able to hold current bank angle)
  - +2 degrees bank angle
  - $a = \begin{bmatrix} -2 & 0 & 2 \end{bmatrix}^T$

\[ a_1: \Delta \varphi = -2^\circ \quad a_2: \Delta \varphi = 0^\circ \quad a_3: \Delta \varphi = 2^\circ \]
Tracking as an RL Problem

- Rewards ($r$)
  - Rewards given to the RL agent are used to update the Q-matrix.
  - Q-Matrix dimensions are ($s \times a$)
    - Current discretization = maximum dimensionality of (114,688 × 3) for stationary targets.

- Reward structure for UAS Tracking problem
  - Target reaching center of image ($r = +20$)
  - Target hitting image boundary ($r = -5$)
  - Target leaving image frame ($r = -20$)
  - Neutral reward for every other situation ($r = 0$)
Episodic Learning

- Episode length and number of episodes are design parameters that must be chosen to make sense for the particular problem.

- For this work, the end of an episode is defined by 2 possible conditions:
  - 500 actions performed (due to time step of 1 sec per action, this is 500 sec)
  - Target leaving the image frame (breaching state constraints)

- When an episode ends, the next episode begins. The initialization of the next episode can be done by:
  - Continuing from the state where the last episode ended
  - Initialize the system to a specified state
  - Initialize the system to a random state (within the bounds)
Environment Modeling and Simulation
Simulation

- **Aircraft**
  - Kinematic Model
    - Position and orientation only
      - $x, y, z$
      - airspeed
      - $\phi, \theta, \psi$
    - Important aircraft specifications
      - Cruising speed
      - Operating altitude

- **Camera**
  - roll, tilt, pan
  - Aspect ratio
  - Zoom
    - FOV angle

MQ-9 Reaper
(MQ-9 Predator B)
Simulation: Assumptions

- Constant radius steady, level, turns
- Discrete jumps in $\phi$
  - Action timespan $>>$ time required to reach commanded $\phi$
- Constant altitude
- Constant velocity
- Fixed camera pose

Balance of Forces

\[
L \cos \phi = W = mg \\
L \sin \phi = a_n = m \psi^2 R_t = m \psi U_1
\]

\[
\tan \phi = \frac{m \psi U_1}{mg} = \frac{\psi U_1}{g} \rightarrow \psi = \frac{g \tan \phi}{U_1} \approx \frac{g}{U_1} \phi
\]
Simulation

- Global simulation space
  - Aircraft pose
  - Target pose

- Learning space
  - Target position in image frame
  - $\varphi$

\[
d = \frac{(X_{\text{img}} - X_{\text{target}}) \cdot \hat{n}_{\text{img}}}{(X_{\text{camera}} - X_{\text{target}}) \cdot \hat{n}_{\text{img}}}
\]
Test Case: Stationary Target

- Camera tilt at -20° from left wing
- Altitude = 152 m
- Target Speed = 0 m/s
- Cruise Speed = 27 m/s
- Reflects 10M learning episodes
Test Case: Stationary Target
Test Case: Linear Moving Target

- Target moves in straight line
- Camera tilt at -20° from left wing
- Altitude = 152 m
- Target Speed = 27 m/s
- Cruise Speed = 27 m/s
- Reflects 5M learning episodes
Test Case: Linear Moving Target
What The Test Cases Show

- Closed-loop tracking control laws, including gains, were developed using an episodic learning process.

- Controller is improved by shaping rewards, and increasing the number of learning episodes.

- Proper representation of actions and rewards is essential for good results.

- Posing the problem to minimize the number of states and actions leads to faster and more efficient learning, faster and more efficient operation.
Test & Evaluation
Wii Remote Infrared Sensor

- Manufacturer: PixArt Imaging
- Resolution: 128x96
- Field of View: ±33° H x ±23° V  
  (determined experimentally)
- Spectrum: 940nm IR
- Refresh Rate: 100 Hz
- Multi-Object Tracking engine
  - 8x subpixel analysis
  - 1024x768 virtual resolution
  - On-line configurable sensitivity
  - Blob tracking for up to four points.
- Interface: 400kHz Fast \( I^2C \)
ConSInt Testbed
Pegasus

- **Features:**
  - Large fuselage internal volume
  - Variable static stability
  - Modular construction

- **Geometry:**
  - Wing Span: 12 ft
  - Wing Area: 18 ft²
  - Length: 10.6 ft

- **Weight:**
  - Empty: 30 lb
  - Maximum Take-Off: 52 lb
  - Payload: 20 lb
  - Fuel: 2 lb

- **Performance:**
  - Maximum Speed: 87 knots
  - Stall Speed (MTOW): 26 knots
  - Endurance: 1 hr
Extensions

- **Fidelity**
  - Improved Image Data & Realistic Camera Characteristics
  - Variable Discretization: Adaptive Action Grid (AAG)

- **Capabilities**
  - Moving Target – Randomized Motion
    - Increases learning state space
    - Complex target movements require more numerous or complex states
  - Momentarily Obscured Target – Tunnels, etc.
  - Multiple Targets – Switching Between Targets On The Fly
  - Non-planar dynamics with velocity as an action (control)

- **Gimballed Camera**
  - Increases action space and learning the state-space
Summary

- Method determines control laws for complex interaction, hard to model, possibly poorly understood systems.

- Excellent for systems where prior engineering knowledge or training data does not exist.

- “Model-free” approach
  - Model not used in synthesizing control laws
  - Model does not appear in the control laws
  - Vehicle/environmental models needed for learning via simulation

- Matured controller has some ability to work on different systems
  - Changes in the original system
  - Similar but different system

- Learning is lifelong
  - The more the controller is used, the “better” it gets
Acknowledgements

This work was sponsored (in part) by the

- Raytheon Intelligence and Information Systems Division under contract C10-00904.
  - Technical Monitor: Michael Moan

  - Technical Monitor: Dr. Fariba Fahroo

Any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of Raytheon or the U.S. Air Force.
Questions
Point of Contact

- Director, Vehicle Systems & Control Laboratory
  John Valasek
  Aerospace Engineering Department
  Texas A&M University
  3141 TAMU
  College Station, TX 77843-3141

  (979) 845-1685
  valasek@tamu.edu

- Web Page
  - http://jungfrau.tamu.edu/valasek
Stationary Target

- States \( s = [X_i \ Y_i \ \phi]^T \)

- Actions \( a = [\Delta \phi \ 0 \Delta \phi \ +\Delta \phi]^T \)

- Goal \( g = [X_{ic} \ Y_{ic} \ \phi]^T \)
Moving Target

- States

\[ s = \begin{bmatrix} X_{i,k} & Y_{i,k} & \phi_k & X_{i,k-1} & Y_{i,k-1} & \phi_{k-1} \end{bmatrix}^T \]

- Actions

\[ a = \begin{bmatrix} -\Delta \phi_k & 0 & \Delta \phi_k \end{bmatrix}^T \]

- Goal

\[ g = \begin{bmatrix} X_{ic} & Y_{ic} & \phi_k & X_{i,k-1} & Y_{i,k-1} & \phi_{k-1} \end{bmatrix}^T \]
Stationary Target with Pan Angle

- **States**
  \[ s = [X_i, Y_i, \phi, \gamma]^T \]

- **Actions**
  \[ a = [-\Delta \phi, 0\Delta \phi, +\Delta \phi, -\Delta \gamma, 0\Delta \gamma, +\Delta \gamma]^T \]

- **Goal**
  \[ g = [X_{ic}, Y_{ic}, \phi, \gamma]^T \]
Moving Target with Pan Angle

- **States**
  \[ s = \begin{bmatrix} X_{i,k} & Y_{i,k} & \varphi_k & \gamma_k & X_{i,k-1} & Y_{i,k-1} & \varphi_{k-1} & \gamma_{k-1} \end{bmatrix}^T \]

- **Actions**
  \[ a = \begin{bmatrix} -\Delta \phi_k & 0 & \Delta \phi_k & + \Delta \phi_k & -\Delta \gamma_k & 0 & \Delta \gamma_k & + \Delta \gamma_k \end{bmatrix}^T \]

- **Goal**
  \[ g = \begin{bmatrix} X_{i,c} & Y_{i,c} & \varphi_k & \gamma_k & X_{i,k-1} & Y_{i,k-1} & \varphi_{k-1} & \gamma_{k-1} \end{bmatrix}^T \]
Temporal Difference and Optimal Control

Chris Watkins’s Q-learning – 1989