Tracking with UAV Using Tangent-Plus-Lyapunov Vector Field Guidance

Hongda Chen[1], K.C. Chang[1], C.S. Agate[2]

[1]George Mason University, [2]Toyon Research Corporation

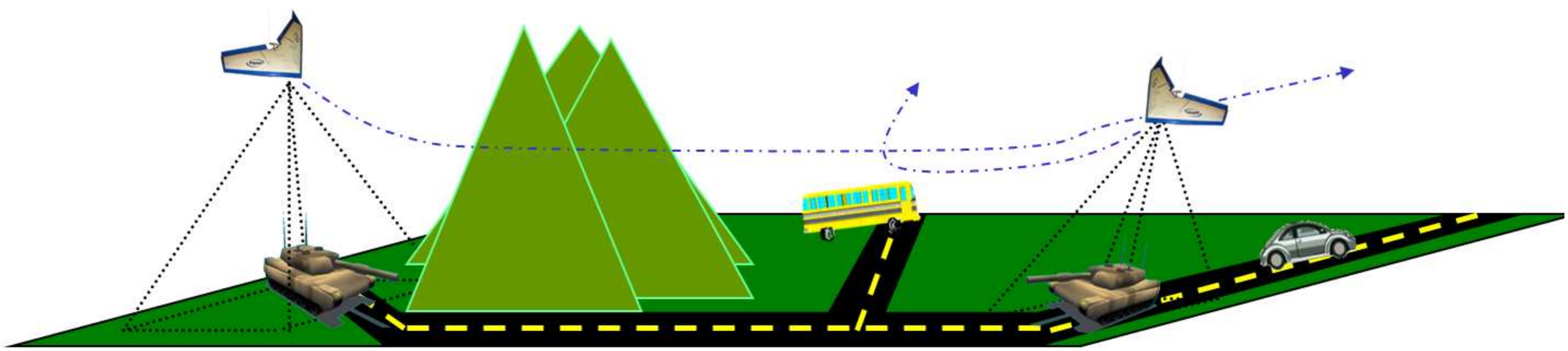
Discussion Outline

- UAV routing problem for ground target tracking
- Minimum flight path solution using Tangent Vector Field Guidance (TVFG)
- Lyapunov Vector Field Guidance (LVFG)
- Exploiting roads and maximizing probability of detection
- Simulation Results
- Summary

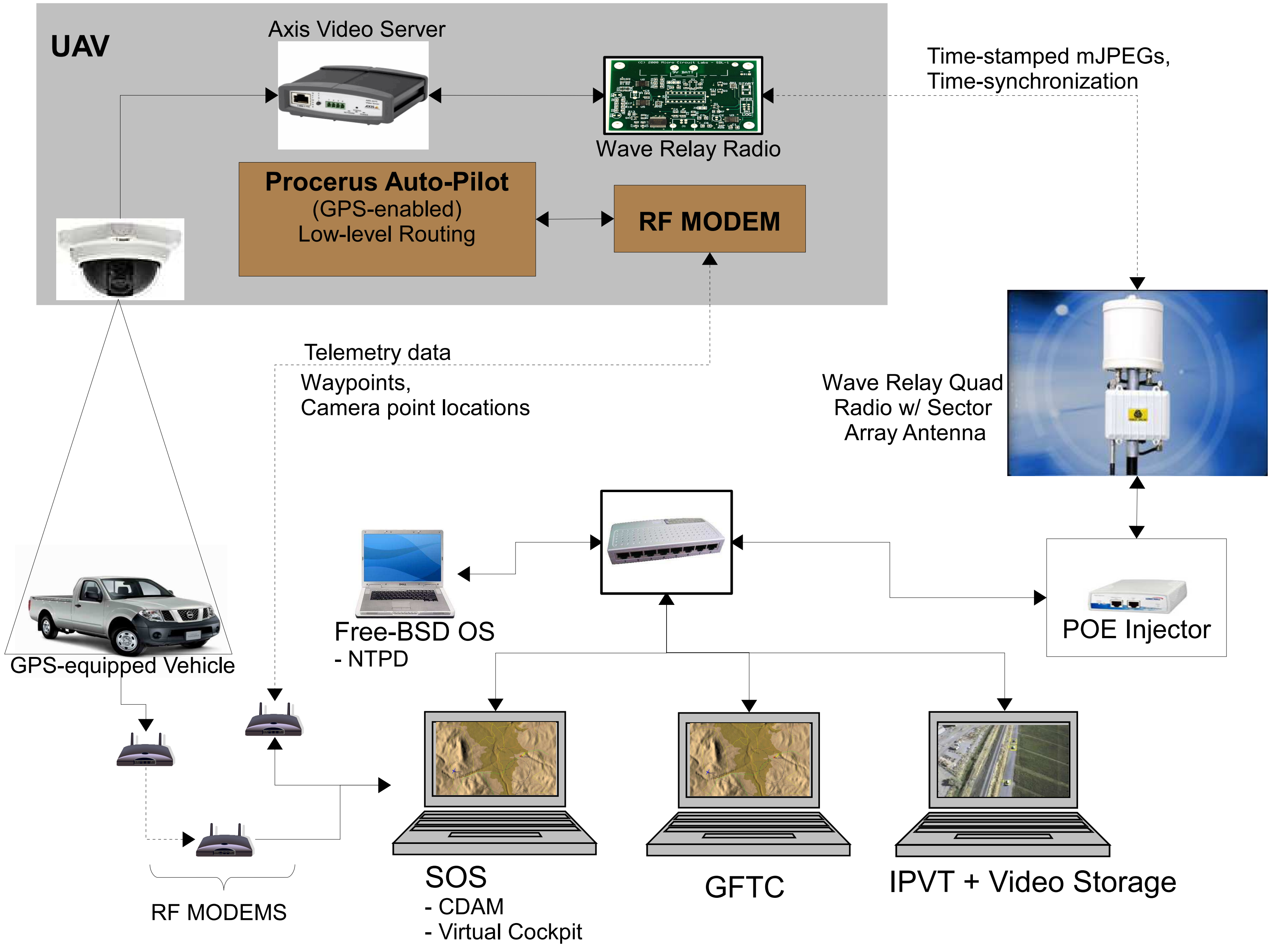
Target Tracking from UAV Video Data

Objective: Track targets-of-interest (TOIs) through surveillance region

- UAV system is cued to follow one or more TOIs
  - UAVs could be cued to a track derived from a standoff asset
  - Operator may “manually" nominate target in video stream
- UAVs have gimbaled EO/IR sensor onboard
- Exploit video to track targets in Geo-registered coordinate system
- “Closed-loop" tracking problem (i.e., tasking and routing essential)



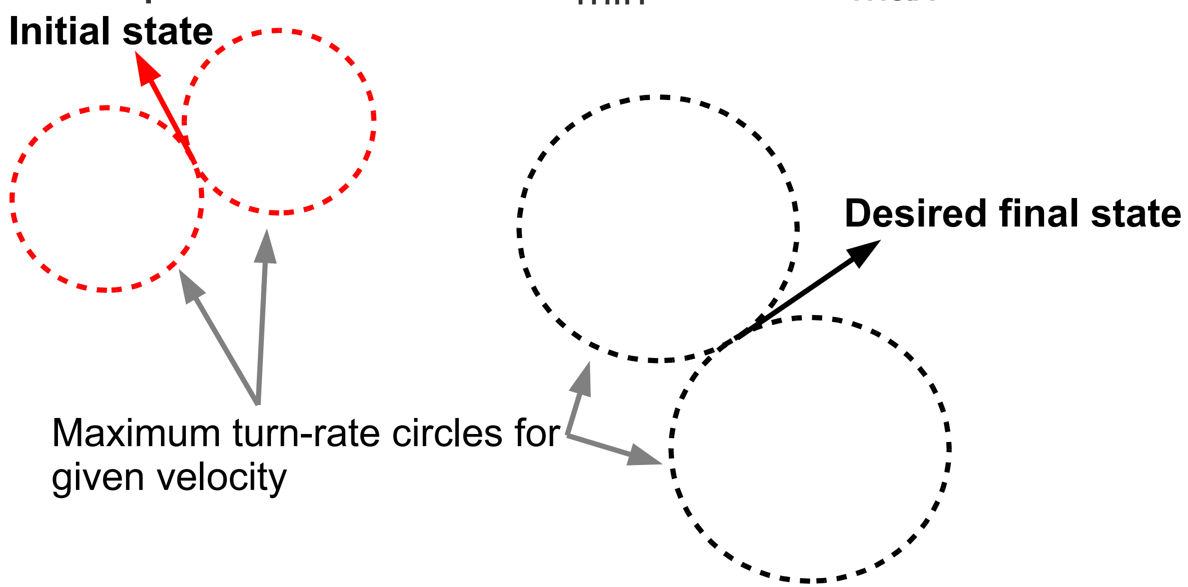
Hardware Setup of UAV System



Initial UAV Control Problem Statement

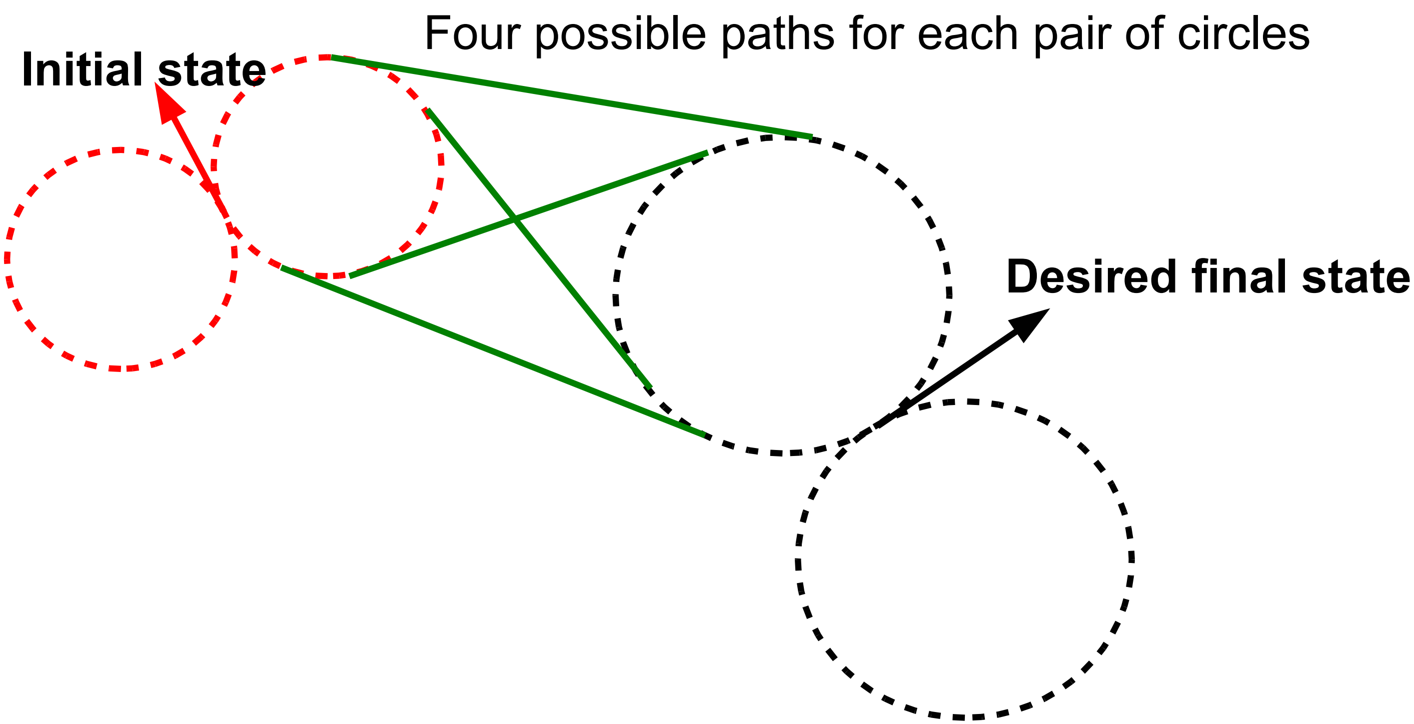
Given a UAV's initial state, $\mathbf{x}(t_{k-1})$, and desired final state, $\mathbf{x}(t_k)$, find the shortest trajectory, $\mathbf{x}(t)\ \ t \in [t_{k-1}, t_k]$, between endpoints.

- The control solution is a set of UAV waypoints and speeds passed to a UAV autopilot system (AP) for low-level control.
- Accurate UAV motion model required to insure AP can achieve waypoints.
- Considered limitations on UAV motion include maximum turn rate, $\omega_{\text{max}} = \frac{v}{R}$ and speed constraint, $v_{\text{min}} \leq v \leq v_{\text{max}}$



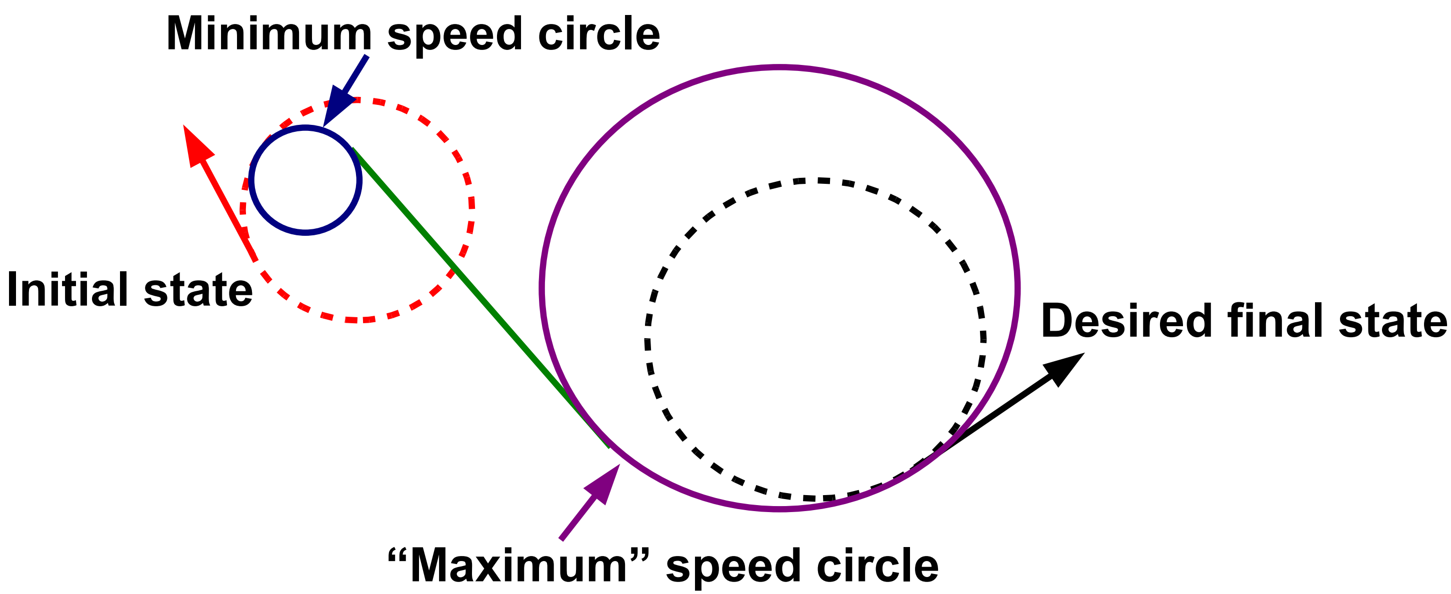
Tangent Vector Field Guidance

- Shortest path involves flying a constant turn rate path (i.e., along a circle) to a tangent line to another constant turn rate path.
- Each state has two (left and right) circular paths
- Each pair of circles has four tangent lines to connect them
- *Algebraic* solution to find best circles and tangent line



Variable UAV Speed Trajectory Solution

- The previous figure assumed (almost) constant UAV speed
- If we consider varying the UAV speed, a shorter distance solution is available
- UAV initially slows to achieve maximum turn rate, then accelerates along tangent line. Finally, UAV decelerates on final circle to achieve desired velocity.



Determining UAV Desired State

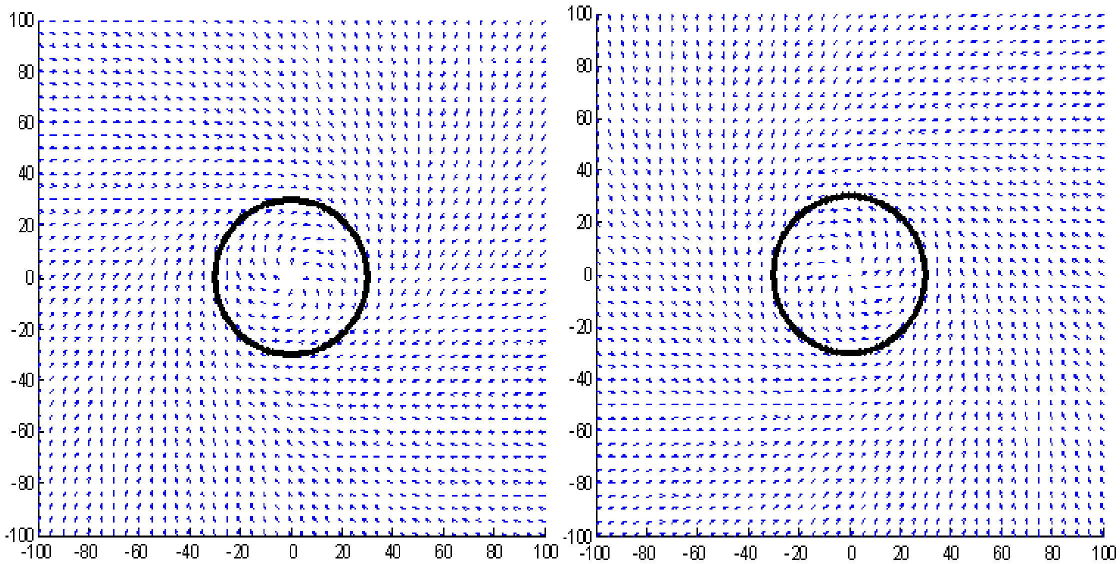
- Given the current UAV state and current target state estimate, we determine approximate location of target at intercept time.
- Desired UAV state (at horizon planning time) chosen based following:
  - Position located on circle of radius R_c about target
  - UAV speed chosen based on target speed (matched, if possible)
  - UAV heading same as target
- Estimated intercept location less accurate when UAV is farther away
- However, the replan interval is shorter than planning horizon (so accuracy of prediction improves as UAV approaches target)
- Does not consider target location uncertainty or probability of detecting target at desired location

UAV Initially within Target Circle

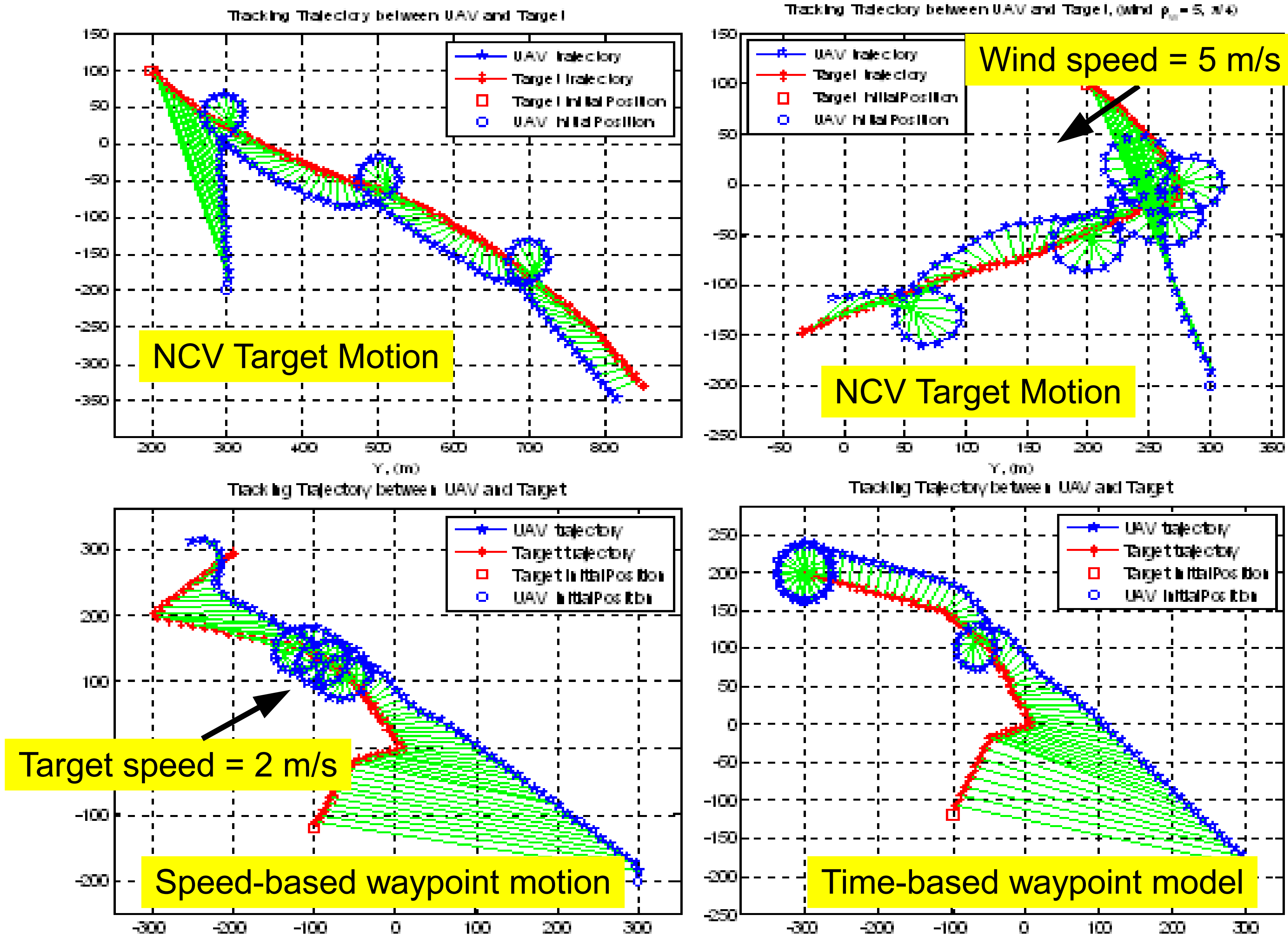
- If the UAV is already within standoff circle, TVFG is not applicable
- When UAV starts in circle, use a Lyapunov Vector Field Guidance Law (Frew et. al. (2008)) based on the following Lyapunov function:

$$\Gamma(x_r, y_r) = \left(r^2 - R_t^2\right)^2 \qquad \text{where } r = \sqrt{(x_r^2 + y_r^2)}$$

- Derive a vector field on velocity vector that leads to circle of desired radius around target for UAV path



Example Simulation Results



Uncertainty in Target's Future Position

- Our goal is to keep the target within the sensor's FOV
- Let $P_D(\mathbf{x}_T(t_k), \mathbf{x}(t_k))$ be the probability of detecting target given UAV position and target position at horizon time
  - $P_D(\cdot)$ can consider sensor resolution and target size, obscuration, etc.
- Choose UAV desired position to maximize target detection

$$\mathbf{x}^\star(k) = \arg\max_{\mathbf{x}(k)} E\{P_D(\mathbf{x}_T(k), \mathbf{x}(k))\}$$

- Target state at t_k is random, so we must maximize *expected* probability of detection with respect to $p(\mathbf{x}_T(k)|Z^{1:k-1})$

$$\mathbf{x}^\star(k) = \arg\max_{\mathbf{x}(k)} \int_{\mathbb{R}^n} P_D(\mathbf{x}_T(k), \mathbf{x}(k)) p(\mathbf{x}_T(k)|Z^{1:k-1}) d\mathbf{x}_T(k)$$

Predicting Target Manuevers

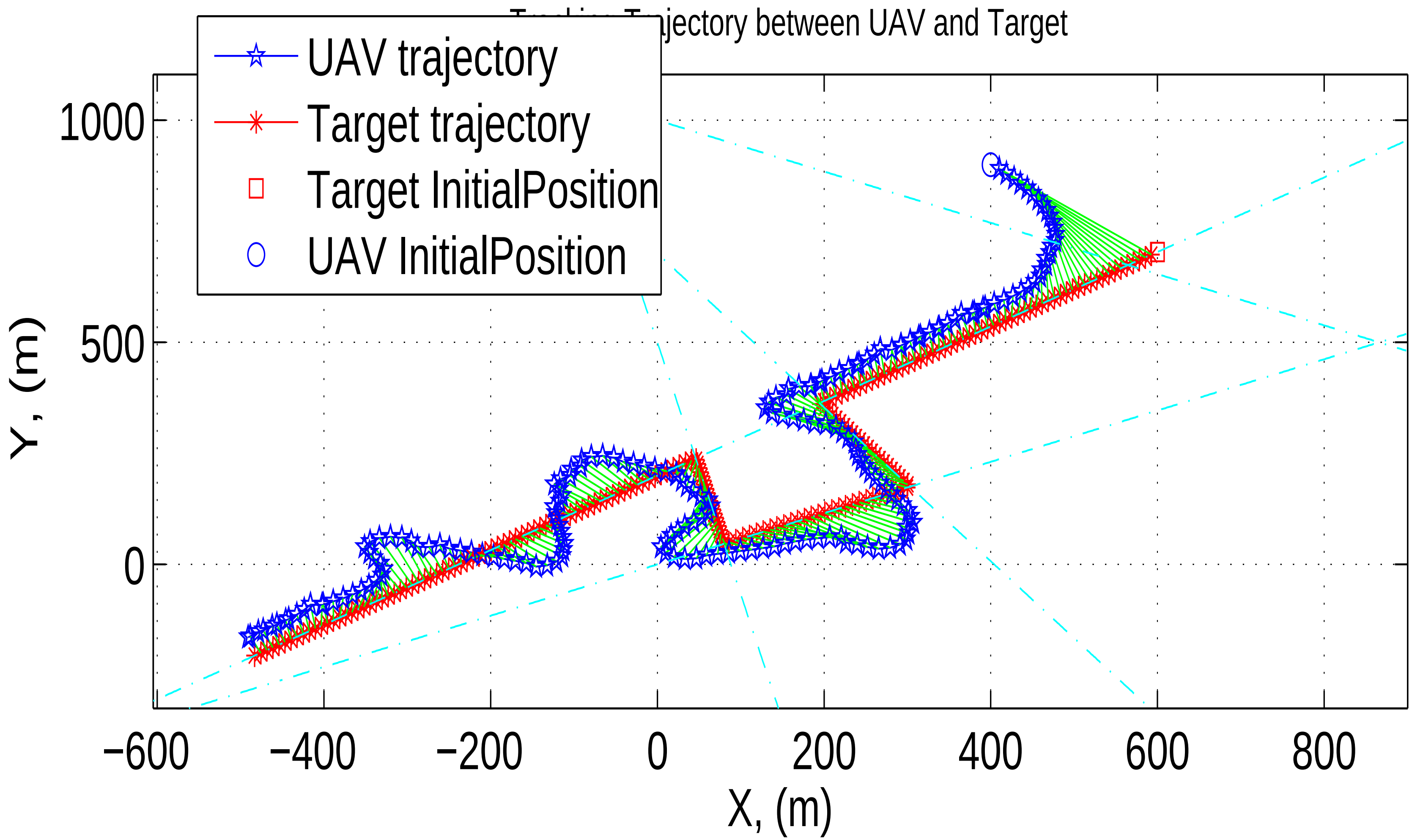
- Target motion prediction greatly improved by exploiting road network information, so we use stochastic sampling of current target state PDF and simulation of target dynamics
- Point mass approximation of predicted target density leads to simple objective function calculation

$$\mathbf{x}^\star(k) = \arg\max_{\mathbf{x}(k)} \sum_i \alpha^{(i)} P_D(\mathbf{x}_T^{(i)}(k), \mathbf{x}(k))$$

- Create grid over reachable UAV states to solve optimization. Let $\mathcal{S}_k$ be the set of reachable UAV states at time t_k, and $\{\mathbf{x}^{(j)}(k)\}_j$ be a set of samples from $\mathcal{S}_k$.
- The desired UAV state, $\mathbf{x}^\star(k)$, is given by

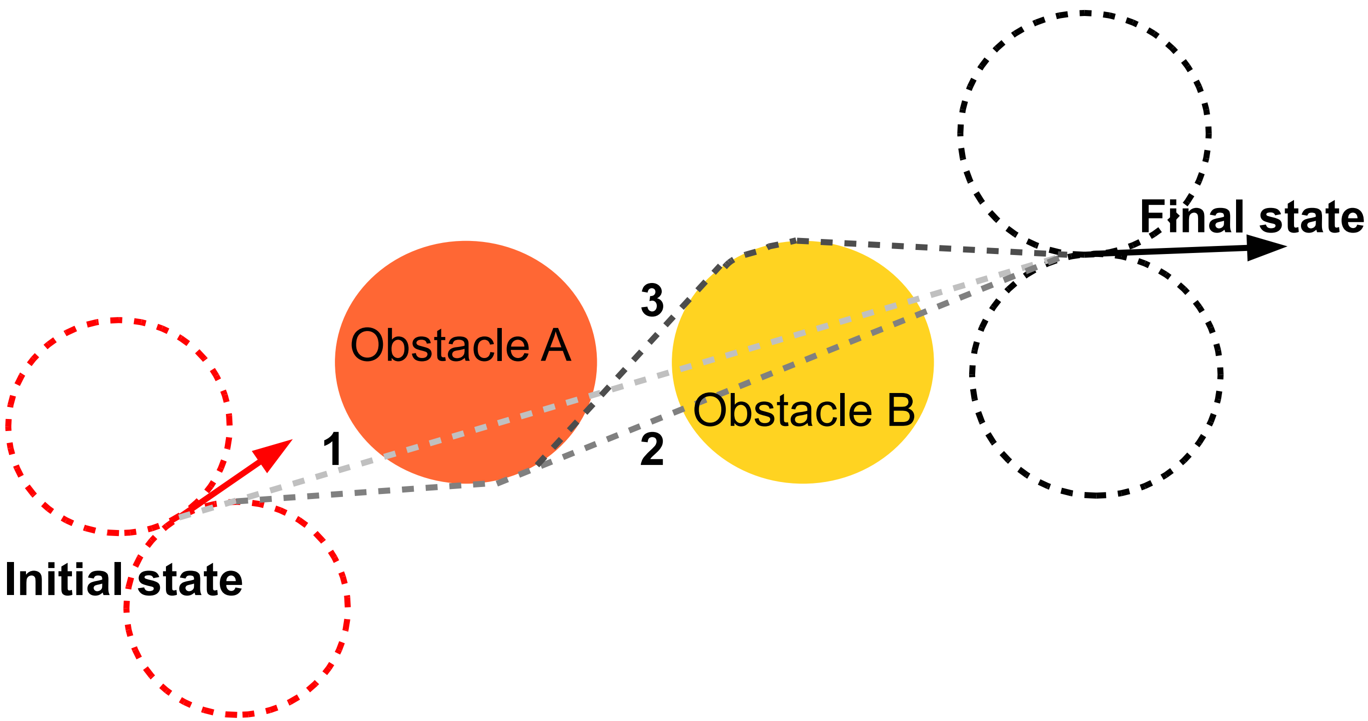
$$\mathbf{x}^\star(k) = \arg\max_{\{\mathbf{x}^{(j)}(k)\}} \sum_i \alpha^{(i)} P_D(\mathbf{x}_T^{(i)}(k), \mathbf{x}^{(j)}(k))$$

Example Simulation Results



Coping with Obstacles and “No-fly" Zones

- Often, there are obstacles restricting UAV flight path
  - Static obstacles such as buildings or no-fly zone
  - Dynamic obstacles due to other aircraft in area
- We model any obstacle as a circular region and use same circle/tangent line approach
- We first consider path to desired position without obstacles. Then determine which obstacles are in the path (start closest to the UAV).



Summary

- We have developed a UAV routing algorithm that combines our TVFG path planning with a LVFG approach (T+LVFG)
  - Can compensate for (constant) wind
  - Approach chooses future UAV position to maximize probability of detecting target
  - Target motion prediction accomplished through stochastic simulation (“particle approach") to exploit road network information
- Obstacles such as No-Fly zones are handled naturally within TVLG approach
- Presented results of algorithm working in Matlab simulation (next step C++ simulation and hardware-in-the-loop system)