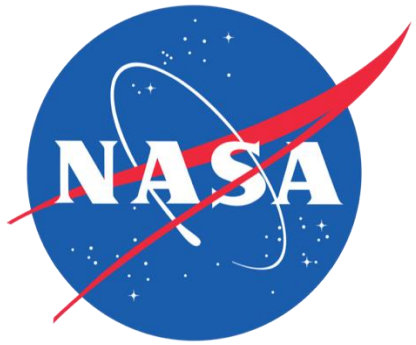
A Verification Framework for Runtime Assurance of Autonomous UAS

J. Tanner Slagel, Lauren M. White, Aaron Dutle,
César A. Muñoz, and Nicholas Crespo

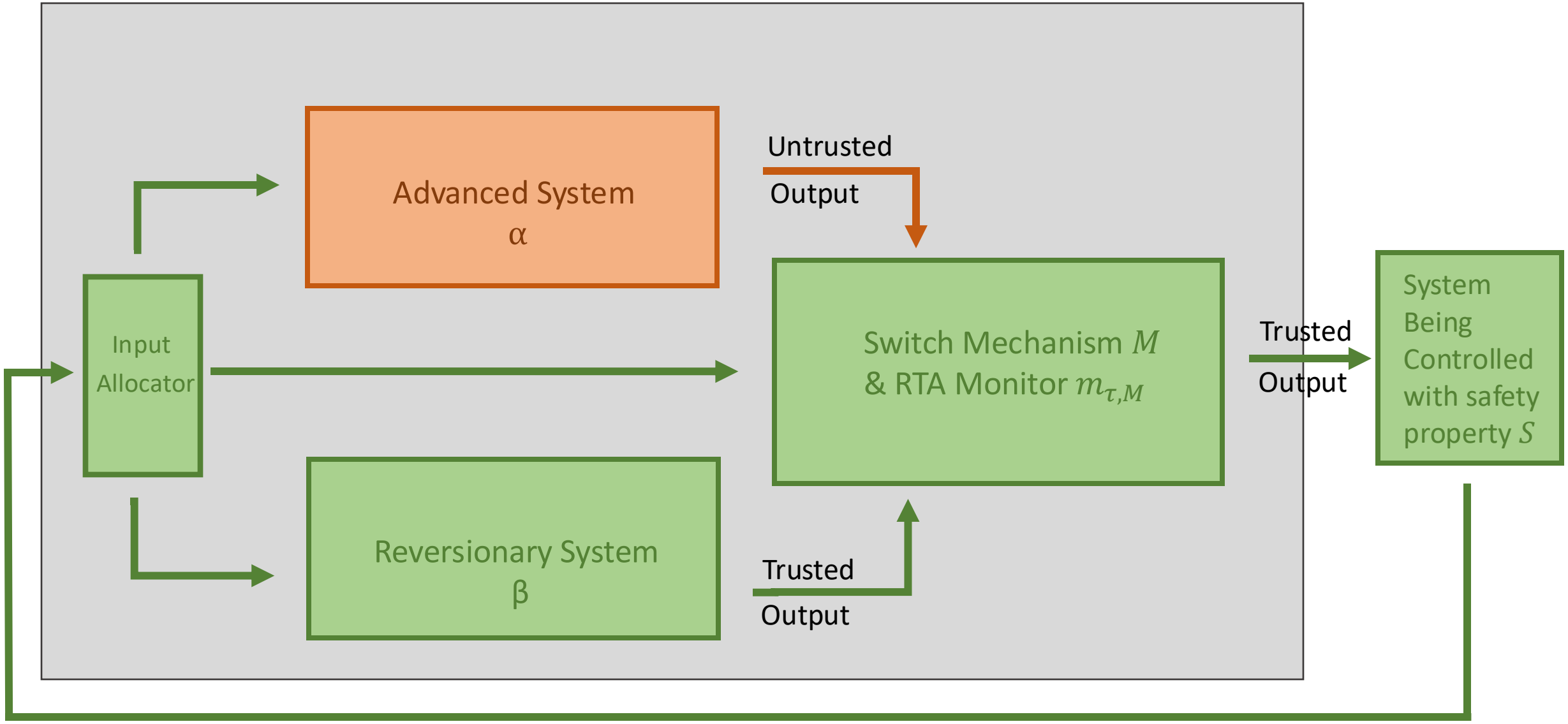
NASA Langley Research Center
Hampton, VA, USA

email: lauren.m.white@nasa.gov



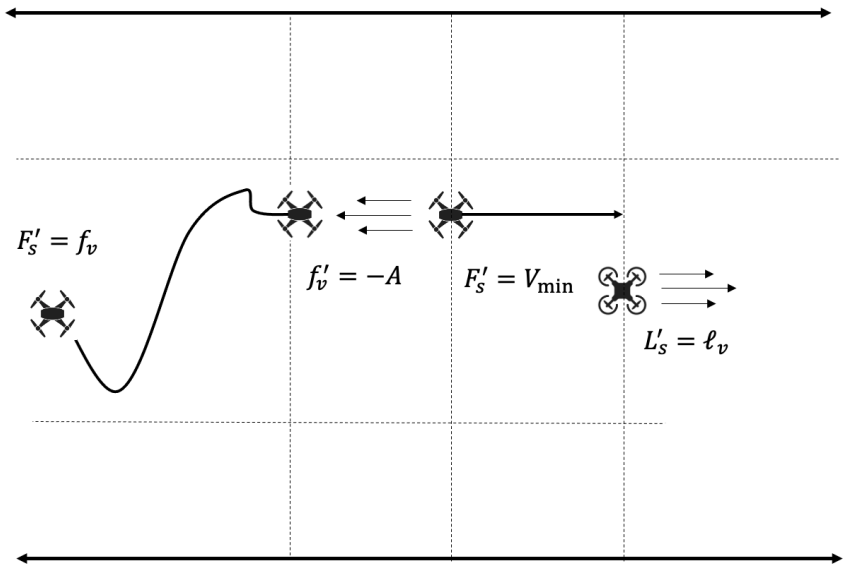
September/October 2024 at the 43rd Digital Avionics Systems Conference (DASC)

Simplex Runtime Assurance (RTA)

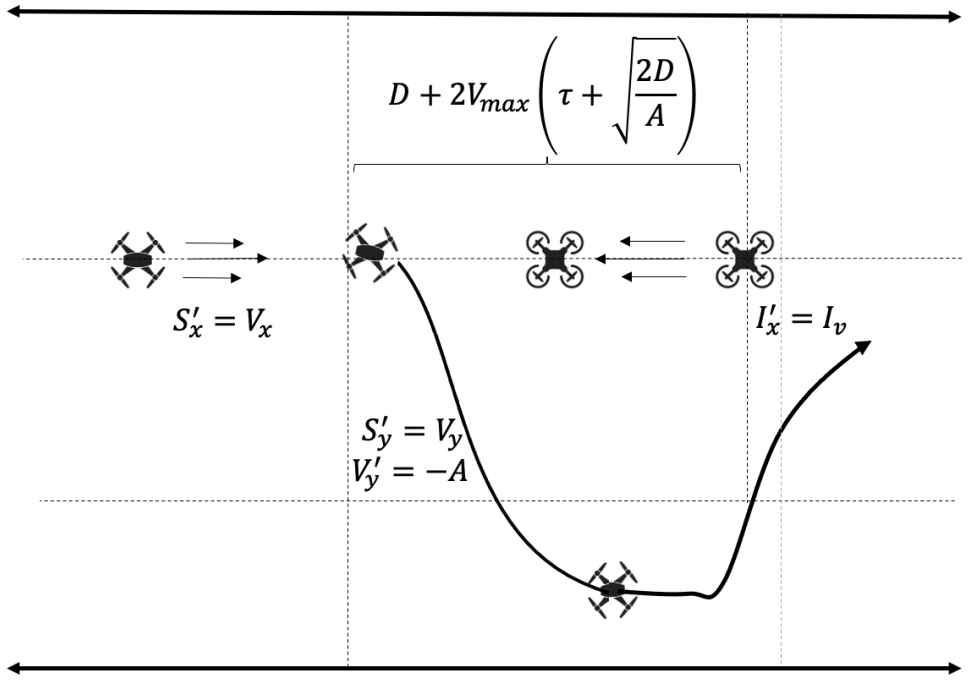


[1]: Runtime Assurance Framework Development For Highly Adaptive Flight Control Systems, SBIR Phase III Report, J.D. Schierman et. al.
https://apps.dtic.mil/dtic/tr/fulltext/u2/1010277.pdf

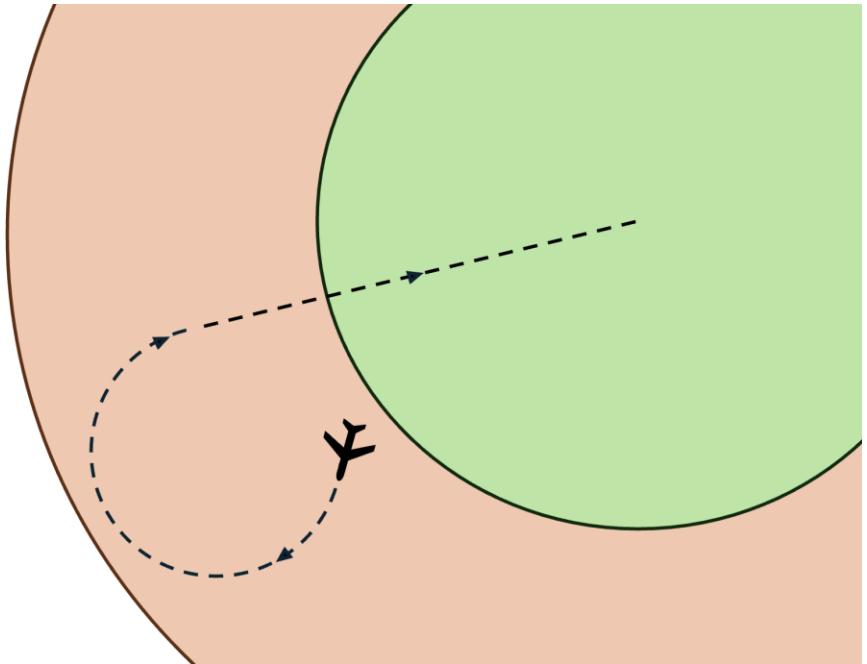
Three examples of UAS operations



1. Follow the leader



2. Productive conflict avoidance



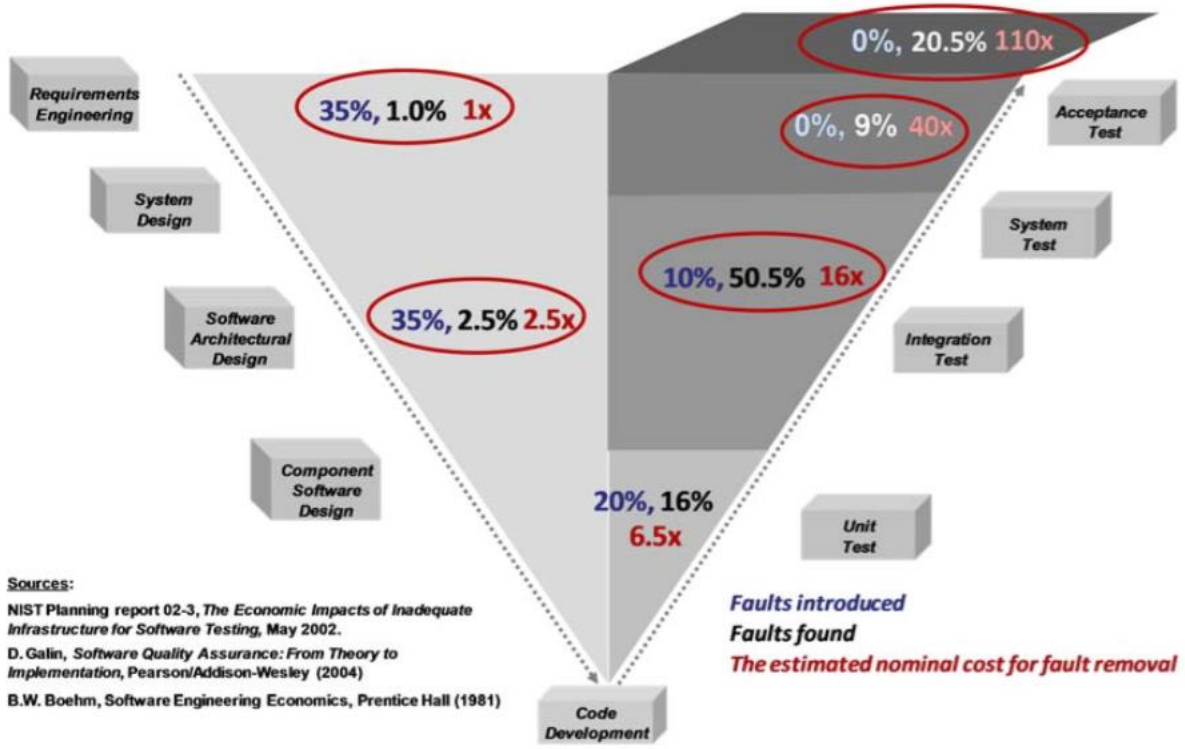
3. Return to safe

What/Why Formal Methods?

What?

Formal Methods refers to mathematically rigorous techniques and tools for the specification, design and verification of software and hardware systems.

Why?

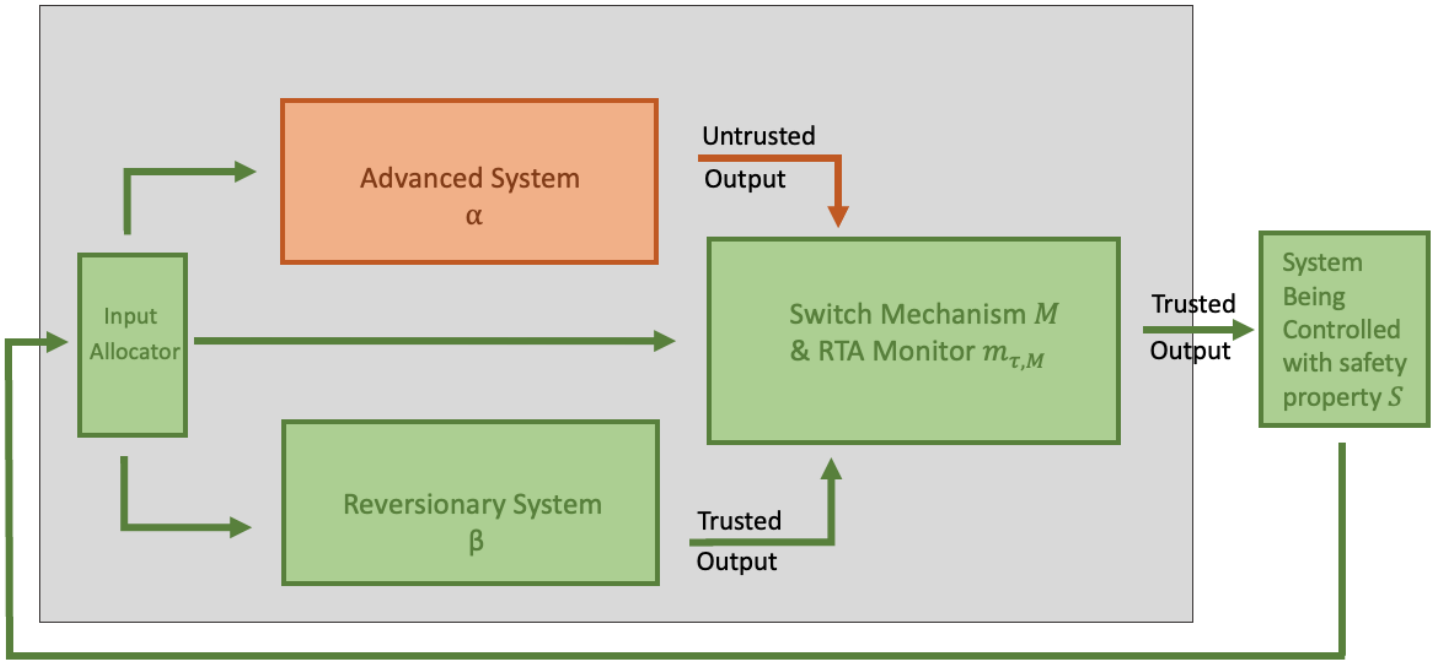


Goal: Use the RTA Simplex framework to formally verify UAS operations where the vehicle is guided by a black box controller.

[2]: NASA Langley Formal Methods Research Program https://shemesh.larc.nasa.gov/fm/
[3]: Reliability validation and improvement framework, Feiler, Peter H., et al. https://insights.sei.cmu.edu/documents/1918/2012_003_001_34081.pdf

Hybrid Programs

$\mathbf{x} := \ell$	Discrete assignments
$\mathbf{x}' := \ell \& P$	Continuous evolution modeled by ODE
$?P$	Test if output satisfies P
$\alpha; \beta$	Sequential execution
$\alpha \cup \beta$	Nondeterministic choice
α^*	Finite repetition of a hybrid program
$\alpha \equiv \beta$	Hybrid program equivalence
$[\alpha]P$	all runs of α ends an a state that satisfies P
$\langle\alpha\rangle P$	some run of α ends an a state that satisfies P
$\Gamma \vdash \Delta$	Γ implies Δ



Define

$$RTA(\alpha, \beta)_{\tau,M} = \left(\left(?M ; m_{\tau,M}(\alpha) \right) \cup \left(?\neg M ; \beta \right) \right)^*$$

While M is true run α with timed monitor **and** While M is not true run β with monitor

Plaidypvs: Differential Dynamic Logic in PVS

dL allows **formal reasoning** of hybrid programs:

- For hybrid program Hp and predicate P
  - All runs $[Hp]\mathrm{P}$
  - Some runs $\langle Hp \rangle \mathrm{P}$

Goal:

$$[RTA(\alpha, \beta)_{\tau,M}]\, S$$

For all runs of the system, the safety property S is satisfied.

Plaidypvs Rules

Plaidypvs rules **simplify** statements in a logically sound way so that the **original statement** is **TRUE** given the collection of **simpler statements** is **TRUE**.

RTA rule in Plaidypvs

$$\frac{\Gamma \vdash S \wedge (M \vee G)\,, \quad S \vdash \left[m_{\tau,M}(\alpha)\right]\left(S \wedge (M \vee G)\right), \quad G \vdash [\beta^*]S}{\Gamma \vdash [RTA(\alpha,\beta)_{\tau,M}]\,S}$$

Where G is a user-instantiated condition that represents a property that carries over when switching from the advanced system to the reversionary system.

RTA rule in Plaidypvs

For a user given property G that carries over when switching from the advanced system to the reversionary system,

$$\Gamma \vdash S \wedge (M \vee G)$$

$$+$$

$$S \vdash [m_{\tau,M}(\alpha)](S \wedge (M \vee G))$$

$$+$$

$$G \vdash [\beta^*]S$$

$$\Gamma \vdash [RTA(\alpha,\beta)_{\tau,M}]\, S$$

IF

the vehicle starts in a safe state,

+

the monitored advanced system controller always ends in a state where the vehicle can recover,

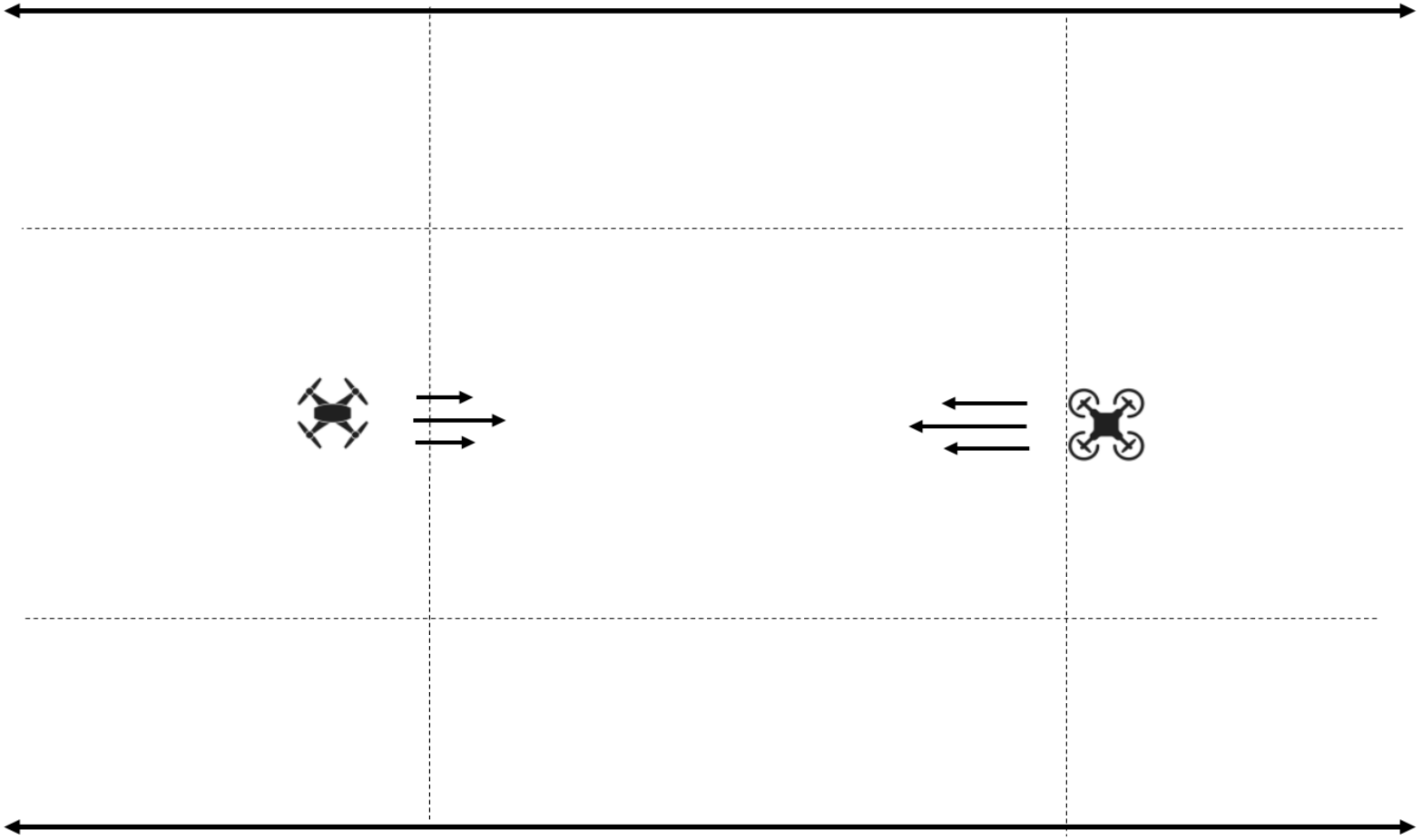
+

and when the vehicle starts in the carry-over state, repeated execution of the reversionary system controller leads to a safe state

THEN

the RTA system satisfies the safety property for every run.

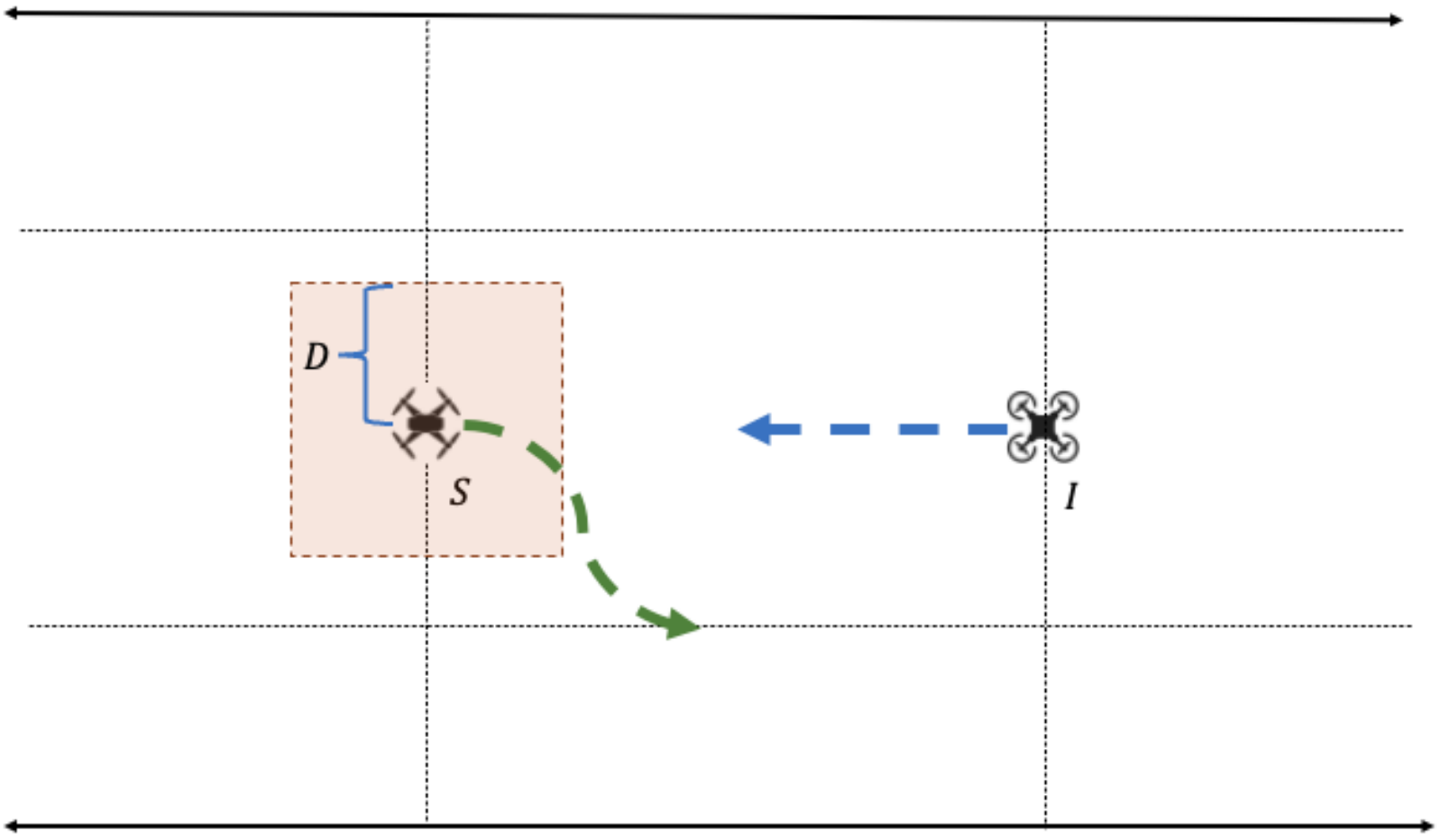
Example: Productive conflict avoidance



Assumptions:

- max velocity V_{max}
- max acceleration A
- sample rate τ

Example: Productive conflict avoidance



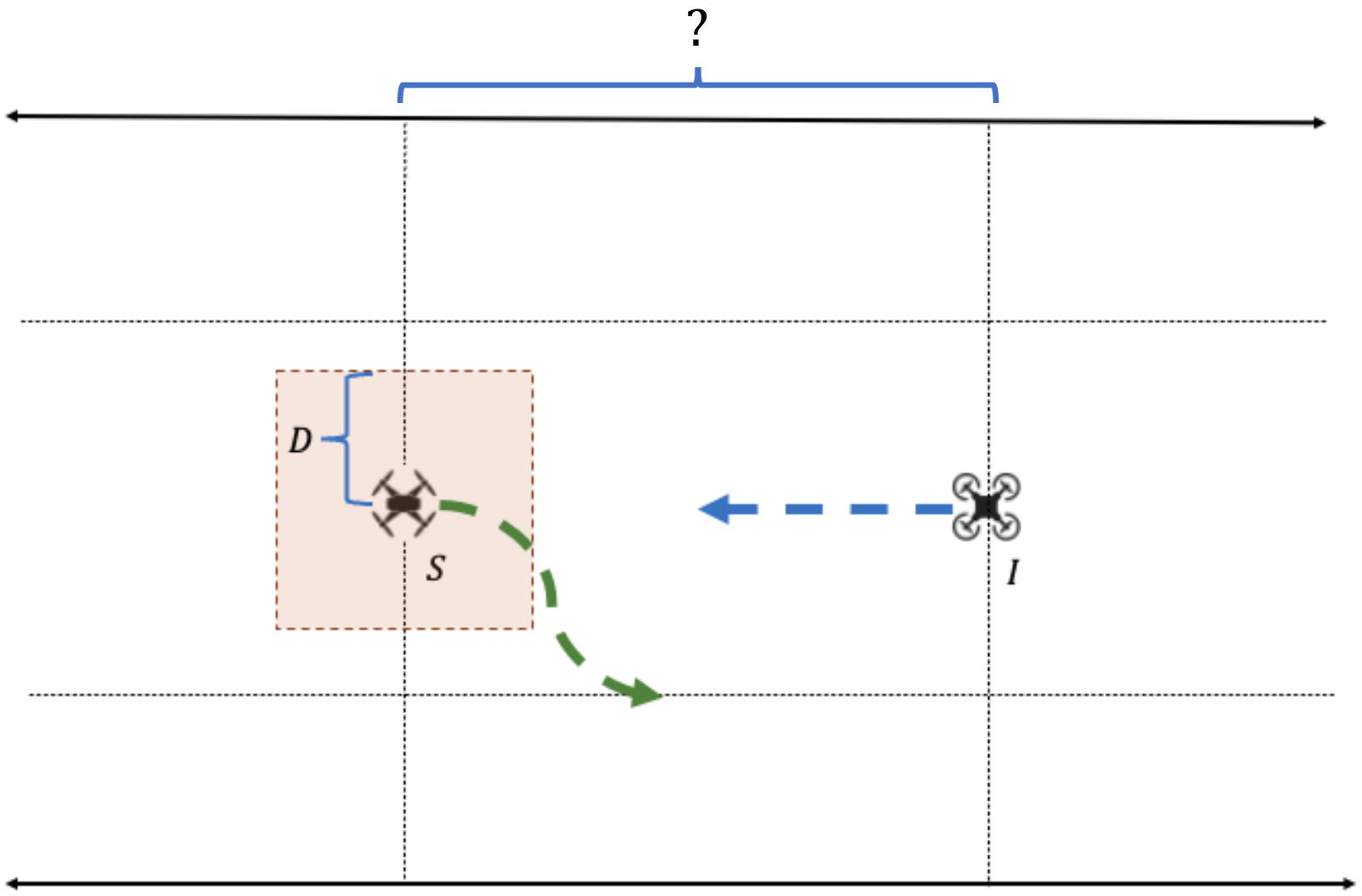
Assumptions:

- max velocity V_{max}
- max acceleration A
- sample rate τ

Safety Requirement:

Distance between I and S is at least D in both coordinate directions.

Example: Productive conflict avoidance



Assumptions:

- max velocity V_{max}
- max acceleration A
- sample rate τ

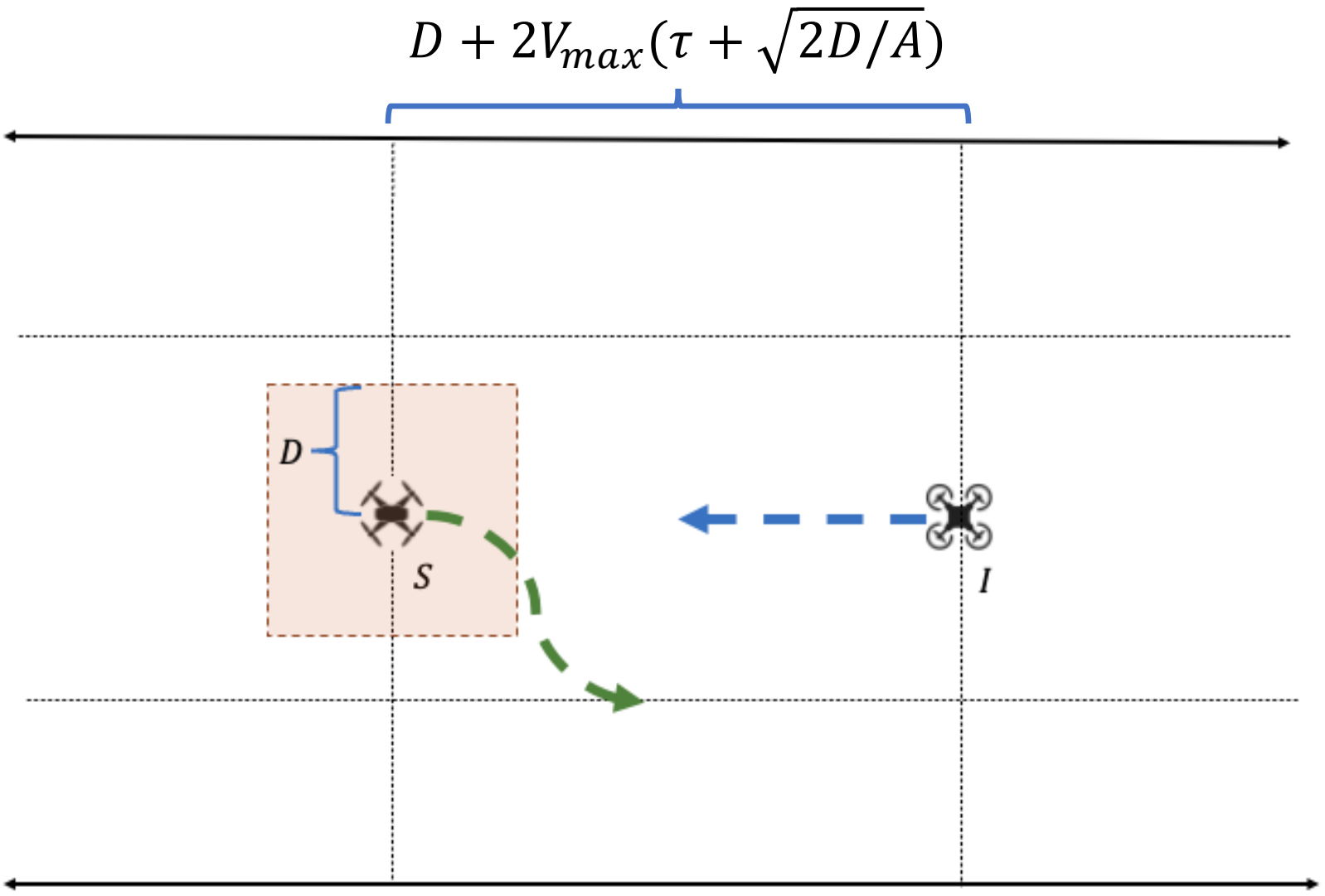
Safety Requirement:

Distance between I and S is at least D in both coordinate directions.

Design Requirement:

What distance apart does the maneuver need to begin to guarantee well-clear?

Example: Productive conflict avoidance



Assumptions:

- max velocity V_{max}
- max acceleration A
- sample rate τ

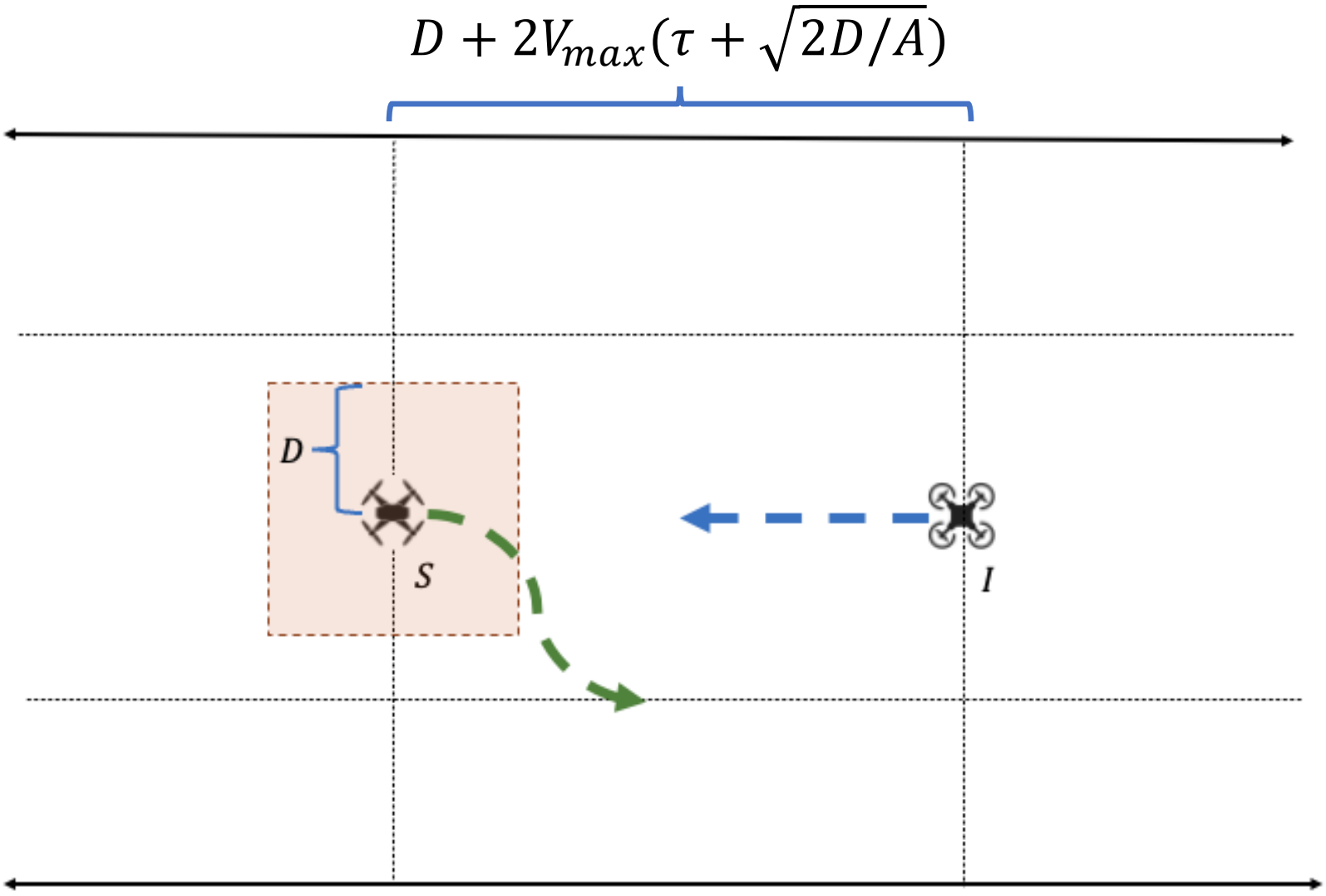
Safety Requirement:

Distance between I and S is at least D in both coordinate directions.

Design Requirement:

What distance apart does the maneuver need to begin to guarantee well-clear?

Example: Productive conflict avoidance



Assumptions

+

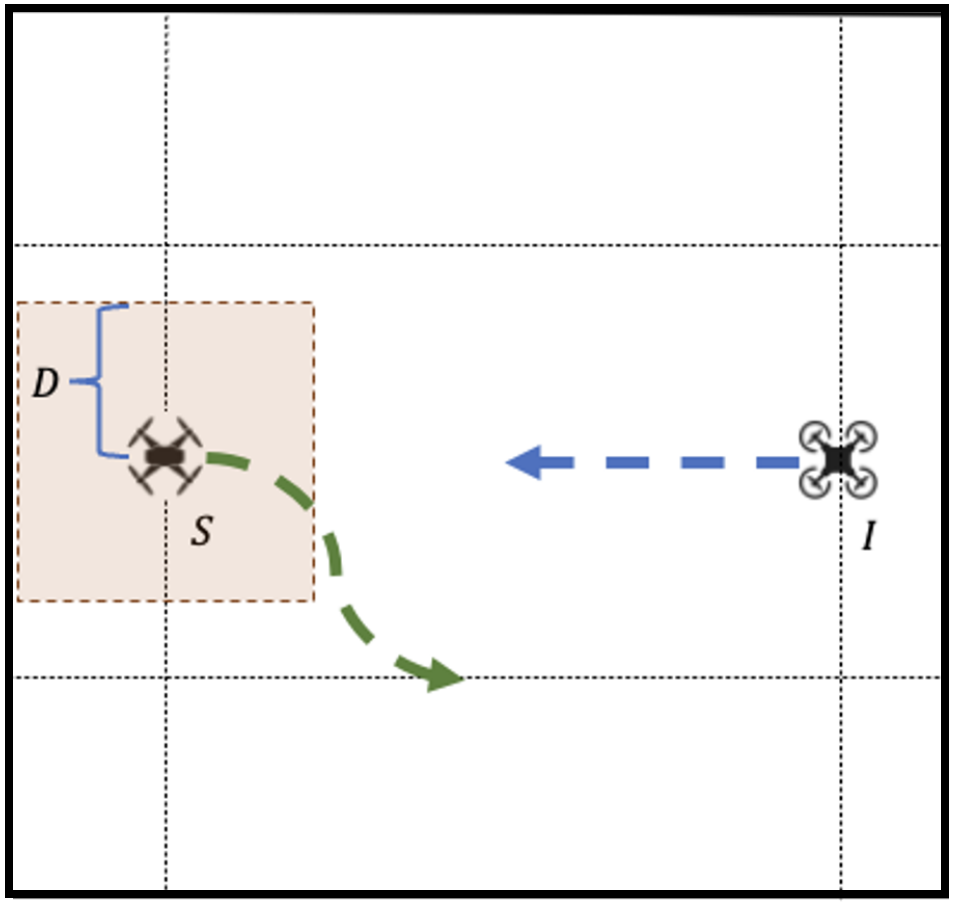
Safety Requirements

+

Plaidypvs verification

Design Requirements

Summary



Plaidypvs can

- model and formally reason about RTA Systems
- help extract and define requirements from the system

Thank you for your attention!

Questions, comments?