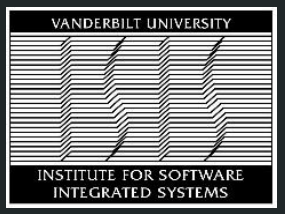


A Testbed to Simulate & Analyze RCPS

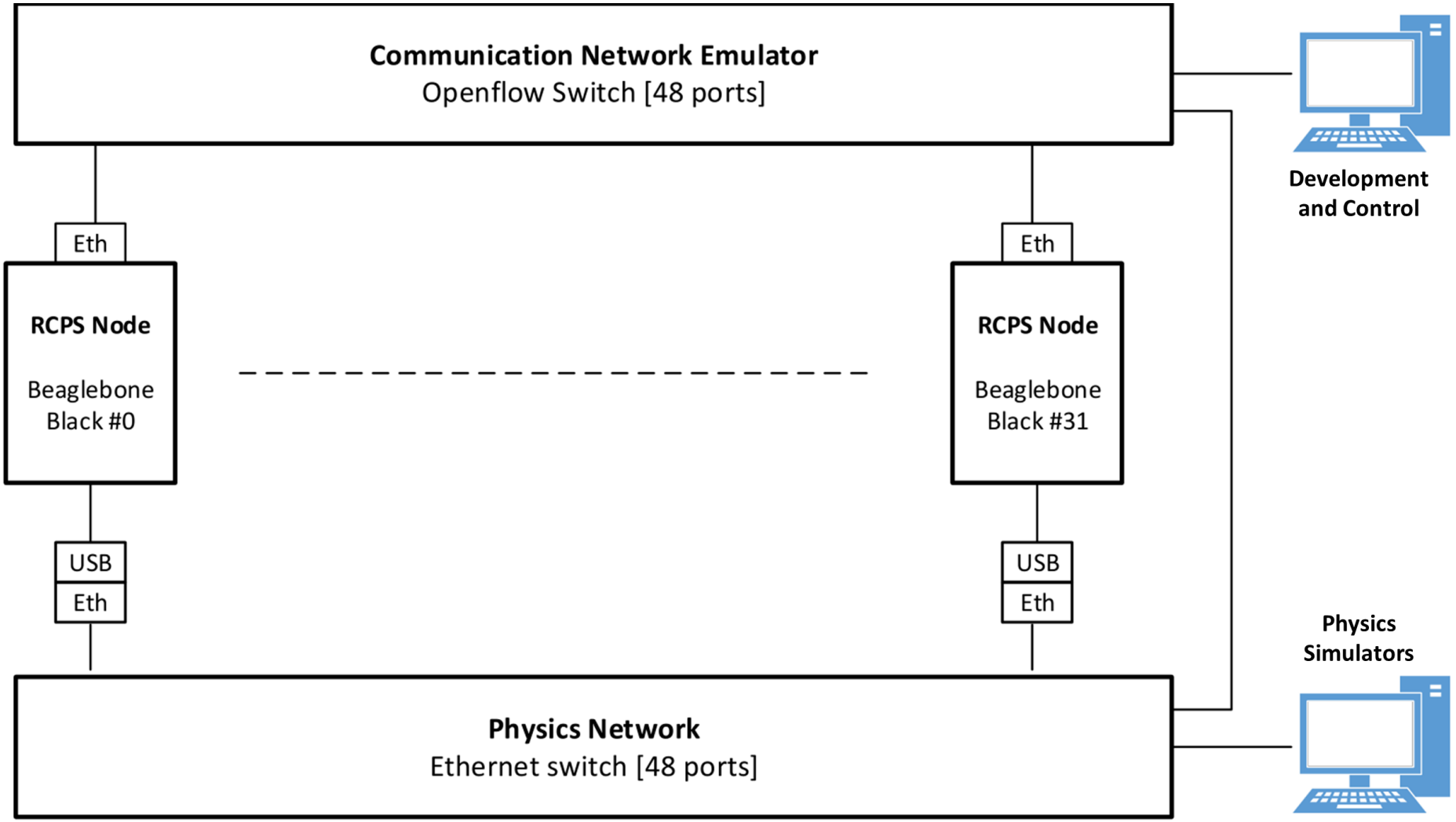
Pranav Srinivas Kumar, William Emfinger, and Gabor Karsai

This work was supported by DARPA under contract NNA11AB14C and USAF/AFRL under Cooperative Agreement FA8750-13-2-0050, and by the National Science Foundation (CNS-1035655). 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 DARPA, USAF/AFRL, or the NSF.

Introduction

- Developing software for Distributed CPS is hard
- Specialized Hardware and Control
- Difficult to obtain public access to implementation details
- Rigorous testing before deployment – **Design-time Guarantees**
  - Performance assurances, reliability, fail-safety etc.
  - UAV/UUV systems, fractionated spacecraft, autonomous vehicles
- Comprehensive testing requires replicating the CPS using a testing platform providing similar hardware and embedded control code as the real system deployment

Testbed Architecture



RCPS Testbed Overview

- *Embedded hardware* running component-based application code – actual code that would run in the real CPS
- *Physical system simulator* that simulates the physics of the physical system, all sensors, all actuators and the environment
  - Code running on the hosts communicates with the physics simulator to get current sensor state and to control the actuators
- Smart network switch (using OpenFlow) allows the emulation of network resources to accurately emulate the system's network
- Integrated analysis and measurement tools for testing CPS apps
- Modeling tools, code generators, and deployment/monitoring utilities mean we spend more time on testing and less time on infrastructure issues

RCPS Testbed Overview

Use Cases

- Security
  - **Algorithms**: Deploy appliations that test various distributed algorithms e.g. consensus algorithms designed to address the Byzantine Generals problem, and measure the speed of convergence of interactions with system faults and resilience counter measures
  - **Infrastructural**: Test different attack vectors for, e.g., gaining unauthorized access to the file systems, bringing down hosts, or compromising nodes.
- Resilience
  - Introduce faults into the system, e.g. sudden host failure, sudden network failure, etc. and measure how the system and its applications respond to the faults
- Analysis
  - Validate analysis models and develop more precise models of interaction patterns for anomaly detection, e.g. DDoS network traffic profiles versus stable system network traffic profiles.

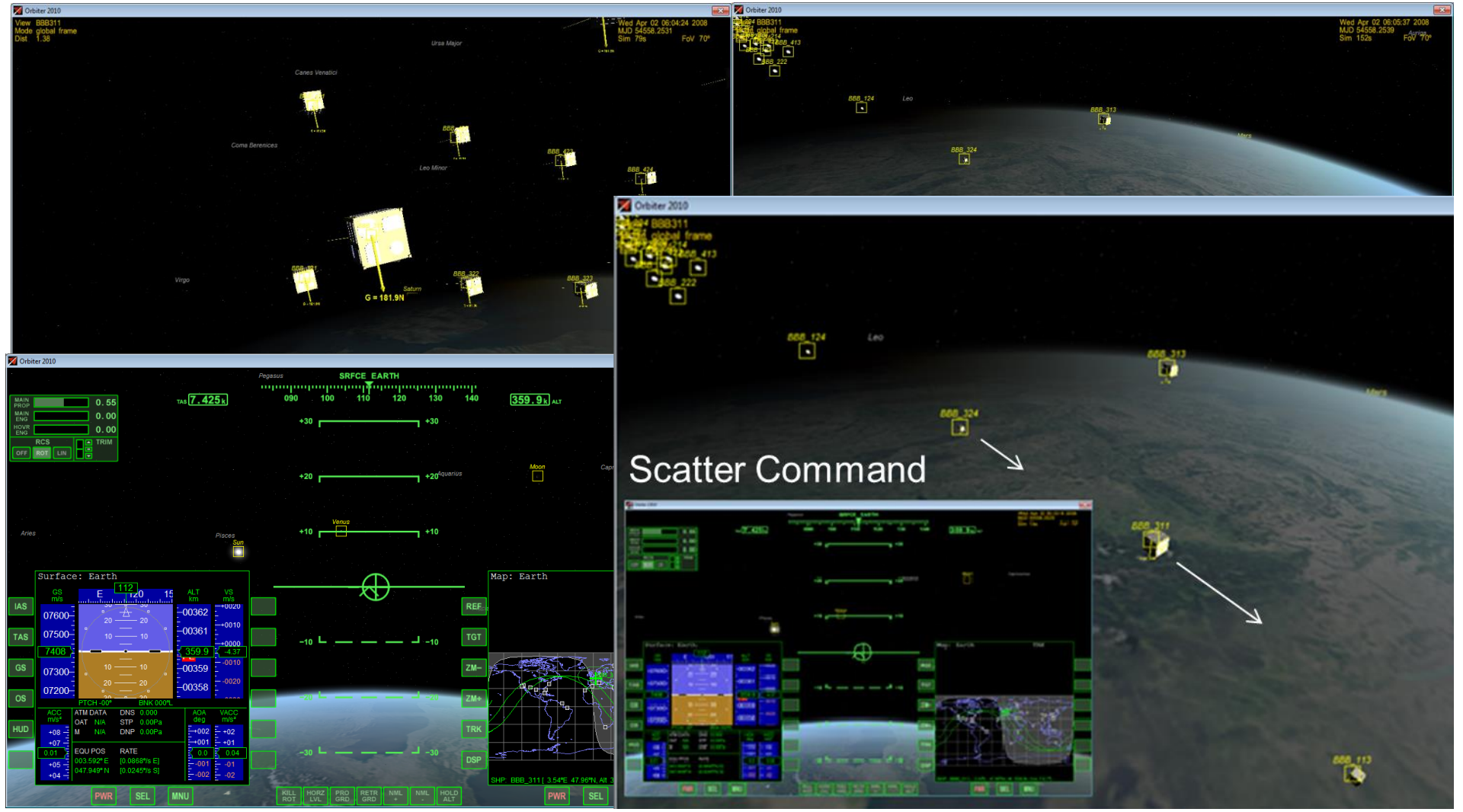
Preliminary Experiments

- Utility in sensing, actuation and control of dynamic CPS scenarios
- Testing Parameters
  - Robot Operating System Middleware
  - CBSE Code Design Principles
  - High Fidelity Physics Simulation Machine with I/O interfaces
  - Development Machine with network access to all RCPS nodes
  - Apps have direct access to physics sim sensors, emulating reality
  - Log real-time activity to aid monitoring infrastructure
- Mission and Safety-critical CPS scenarios
  - Orbiter Space Flight Simulator
  - Kerbal Space Program
  - SUMO

Orbiter Space Flight Simulator

- We have 32 RCPS nodes running component-based flight software
  - **Cluster Flight Control**: handles cluster trajectory and management
  - **Satellite Flight Control**: Retrieves satellite state from sensors, actuates satellite thrusters and communicates internal state to all other satellites
  - **Wide-area Monitoring**: Imaging applications with distributed camera sensors to produce processed images
- RCPS nodes maintain coordinated cluster flight
- Cluster scatters on reception of an emergency *scatter* command
- Applications run on our testbed under **ROS** integrating with **Orbiter**
  - Send scatter command at random intervals to cluster leader
  - Dissemitate command to rest of cluster
  - Satellites scatter – seen in Orbiter

Orbiter Space Flight Simulator

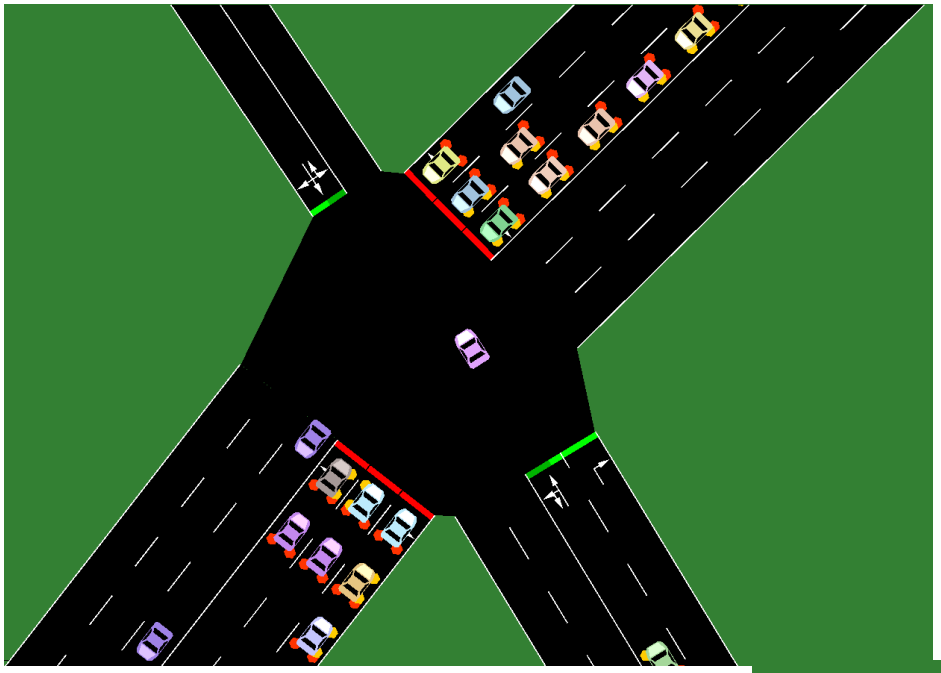


Kerbal Space Program – Autonomy & Control

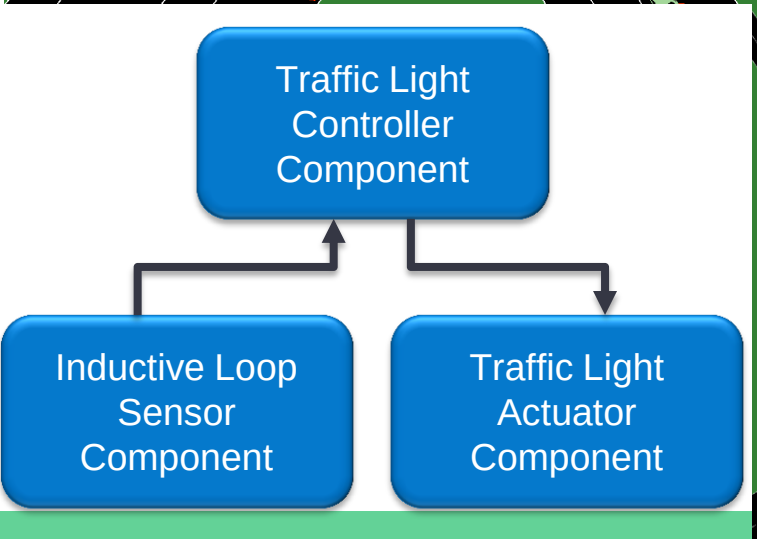
- Widely popular space flight simulator
- Aerodynamic, Gravitational & Rigid body Simulations
- Every man-made object follows Newtonian Dynamics
- Rocket thrust and aerodynamic forces are accurately applied to the vehicles based on directions and precise positions in which the force-affected elements are mounted
- Autonomy & Control Testing:
  - Flight Control – Takeoff, cruise control, and land
  - Self-driving car – Follow waypointing algorithm

Kerbal Space Program – Autonomy & Control

SUMO – Resilient Sensing to Cyber-Attacks



- Deploy sensors, actuators, and controllers on the network
- Attack specific groups or the system overall with e.g. DDoS
- Analyze how the application and system responded to the attacks
  - Develop optimal sensor/actuator placement and network configuration for resiliency against faults and attacks



Conclusions

- The RCPS testbed provides a HIL testing facility for experimentation with CPS code and system prototypes against real-time physics simulator(s) and an embedded network
- The testbed is supported by a model-based development environment for the rapid construction and deployment of CPS application software
- The flexibility of the physics simulation and the network emulation enables the execution of complex testing and evaluation scenarios