Predicting Performance

The Story of Rocket Propellants, Software Ports, Joysticks at Work, and the Slinging of Data Over Networks

Chris Frost

Mentor: Jason Rupert

About Chris Frost

- School: The University of Virginia, Upcoming Second Year
- Majors: Computer Science and Mathematics
- Department: Missile Systems (2nd year)

- Other Academic Interests: Engineering, Physics, and Cognitive Science
- Non-academic Interests: Running

Outline

- *Geometry Tester*
- Rocket and DATCOM Ports
- JMASS, Joysticks, and Simulation Viewers, Oh My!

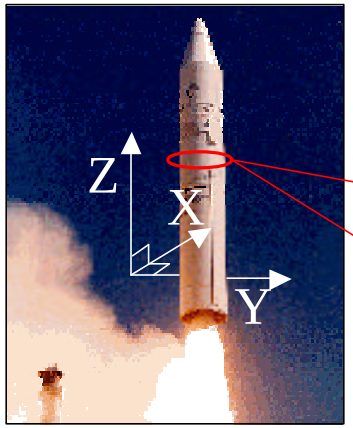
Geometry Tester

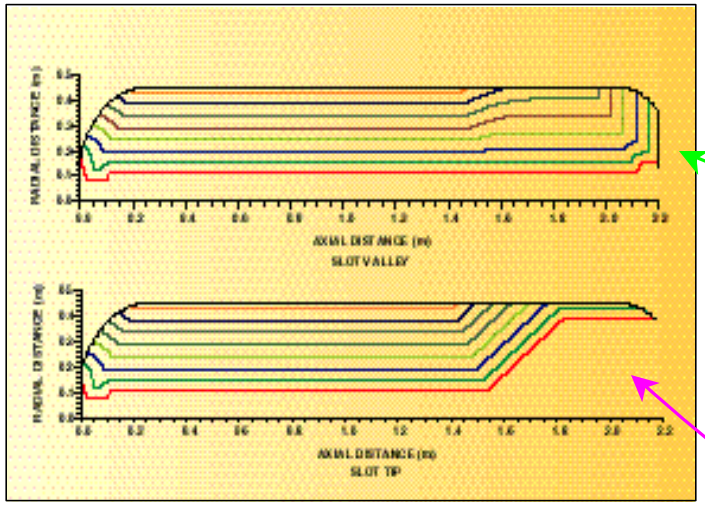
- Problem: Reverse engineering solid rocket propellant geometries is very time consuming
- Goal: Streamline and automate this task

Geometry Tester: Background

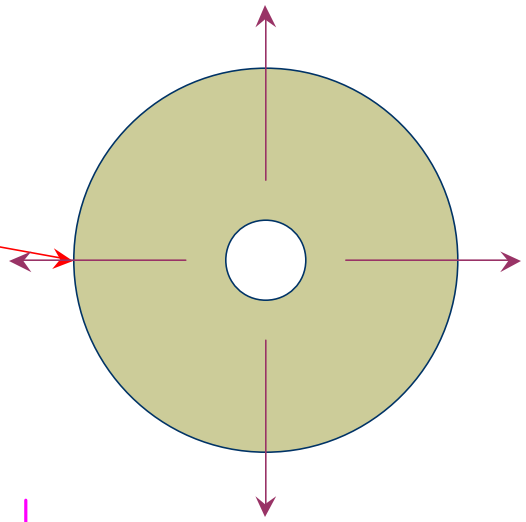
- Explanation of solid propellant shapes and their effects on time vs chamber pressure

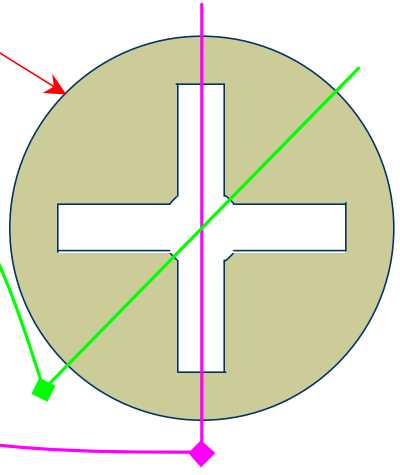
Geometry Tester Background: Solid Propellant Geometry



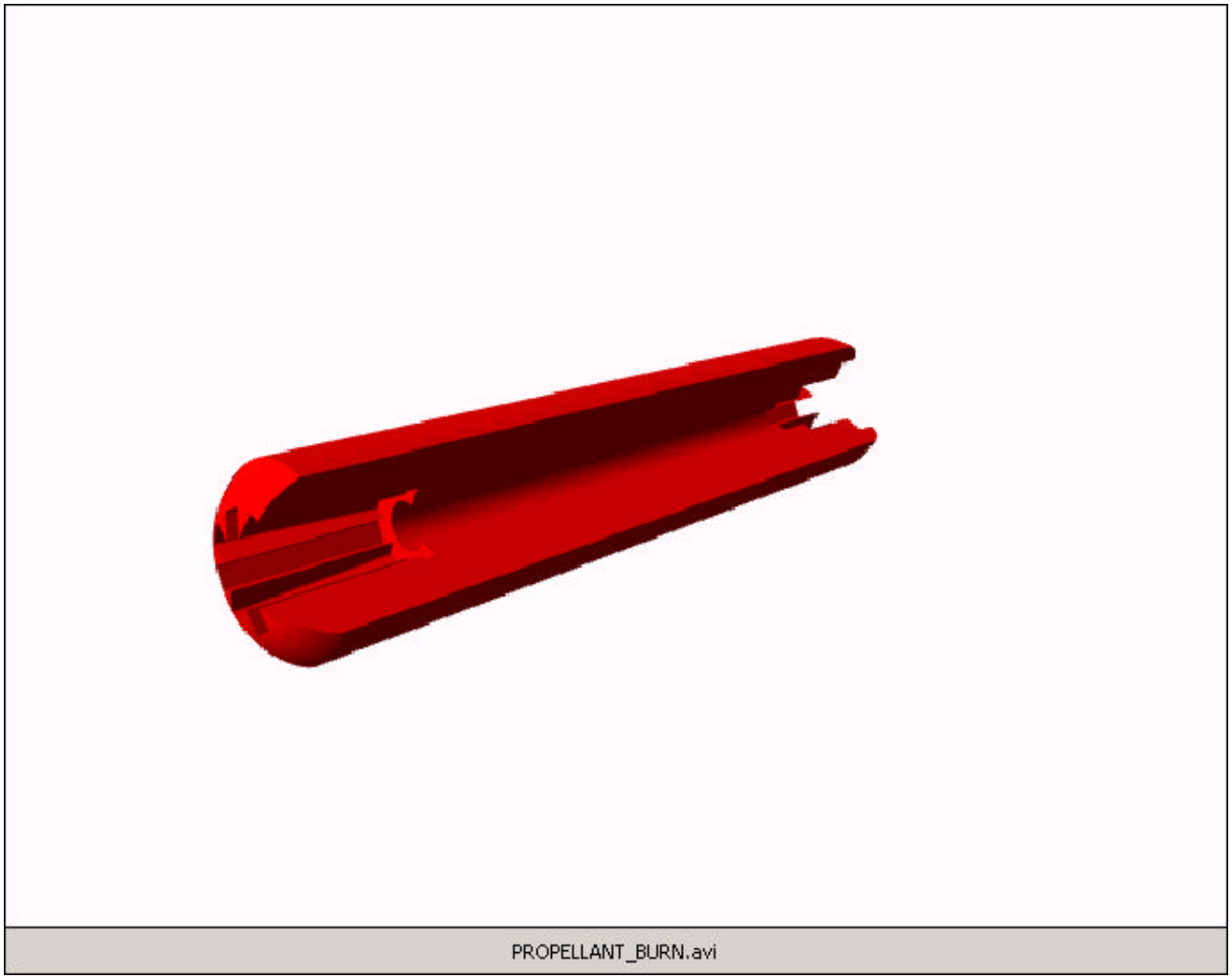


Side Views





Geometry Tester Background:



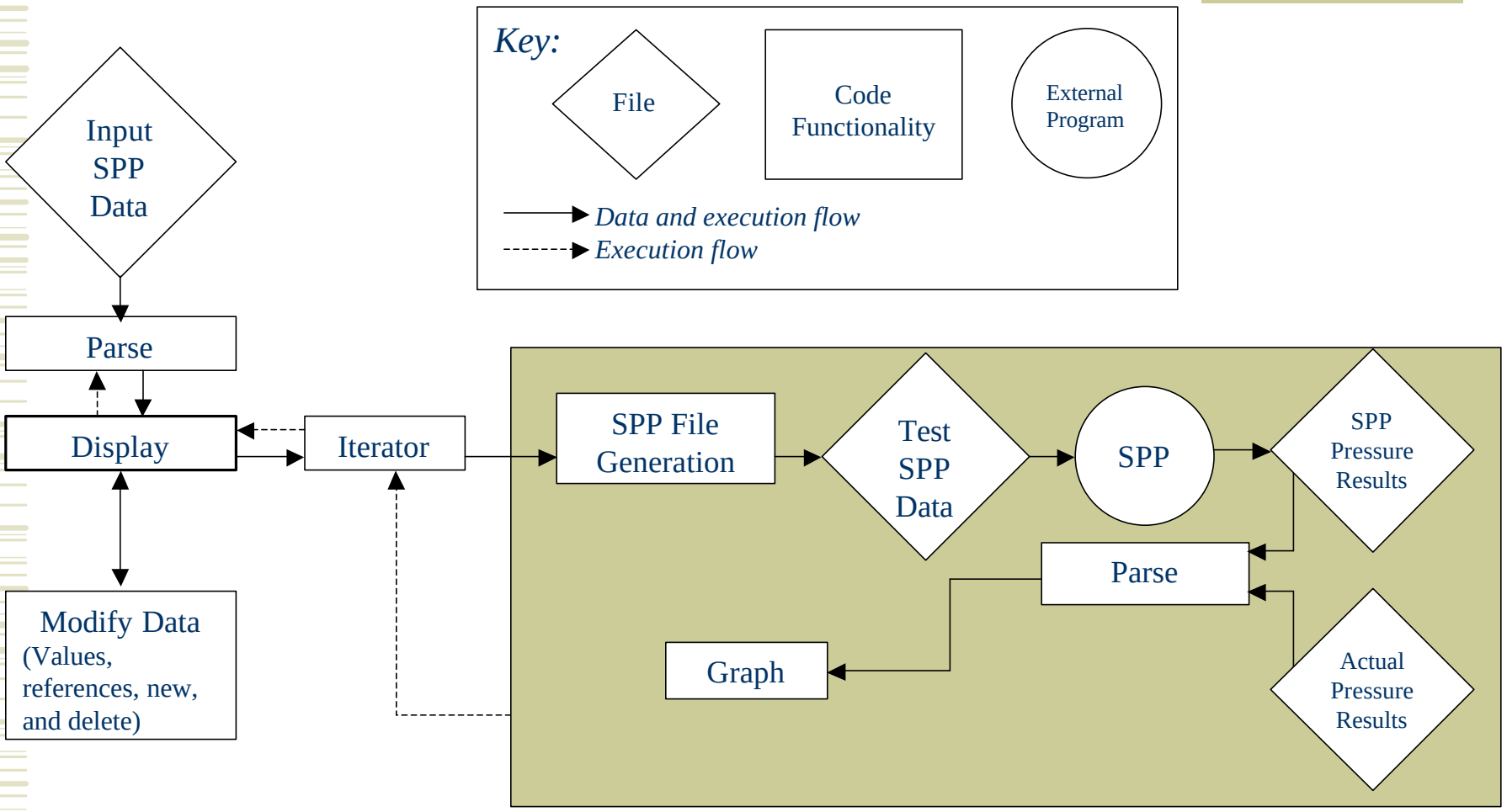
Geometry Tester: Background

- Explanation of solid propellant shapes
- Purpose of matching time vs pressure:
  - Allows us to find a geometry providing similar thrust characteristics
  - Can then simulate or build a rocket with the same propulsive characteristics
- Solid Propellant Program (SPP): Performance Predictions

Geometry Tester: Capabilities

- Read and write SPP files
- Read pressure data files
- Display and modify numerical and symbolic geometry data
- Create and delete objects and records
- Create plots comparing time vs pressure

Geometry Tester: Program Flow



Geometry Tester: Main Window Screenshot

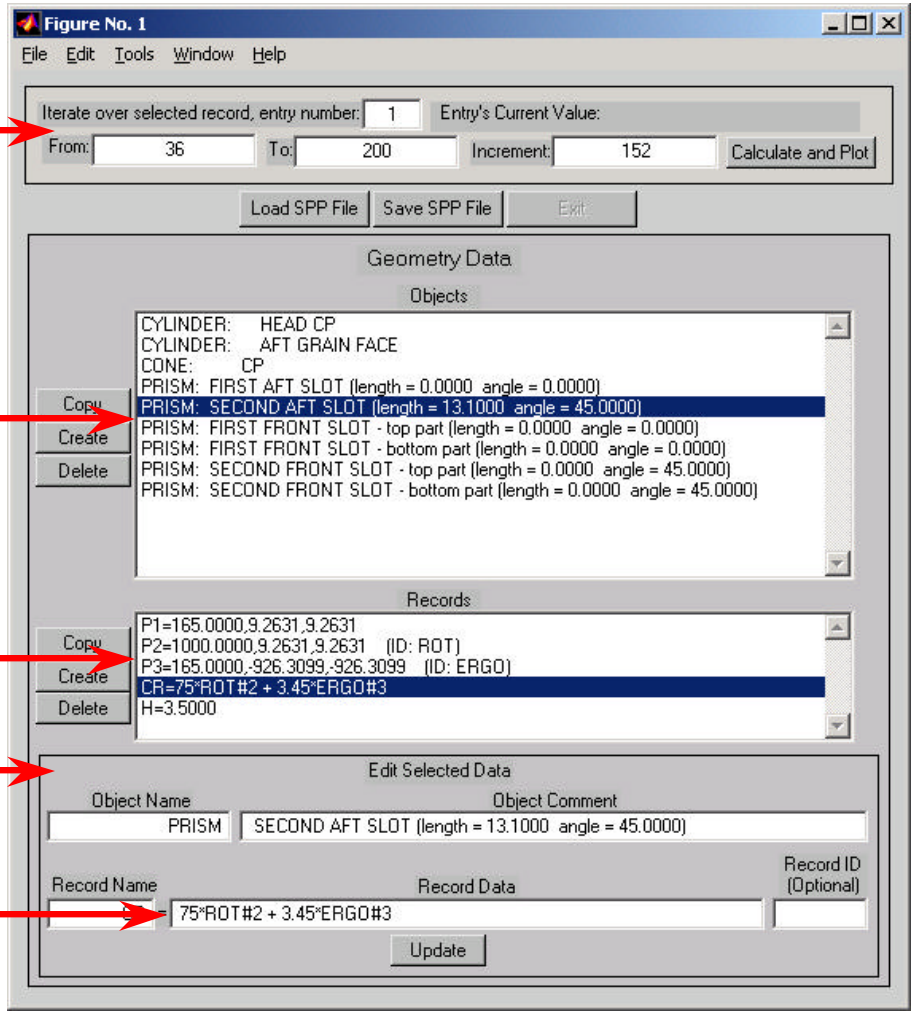
Iteration Data

List of Objects

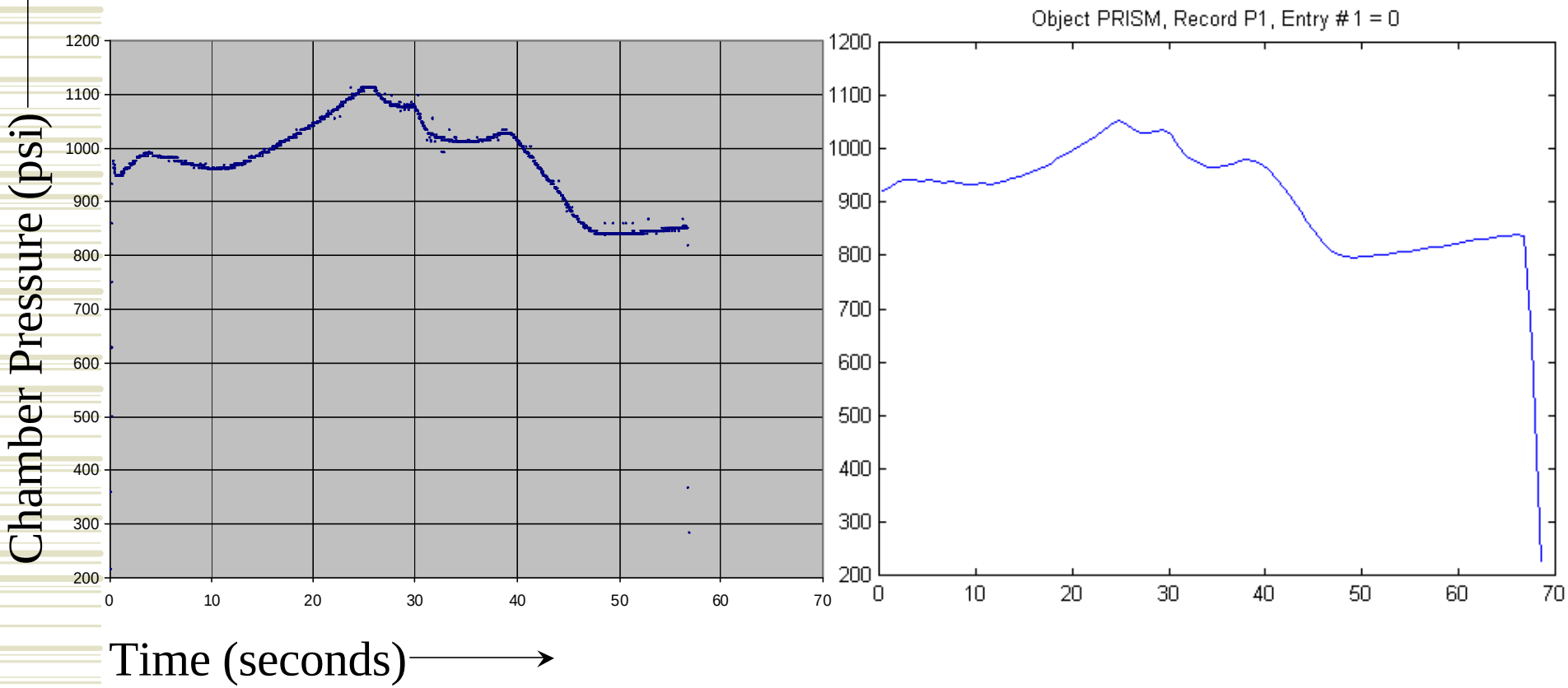
Object Parameters

Entry Data

Equation



Geometry Tester: Example Model



Outline

- Geometry Tester
- *Rocket and DATCOM Ports*
- JMASS, Joysticks, and Simulation Viewers, Oh My!

Rocket and DATCOM Ports

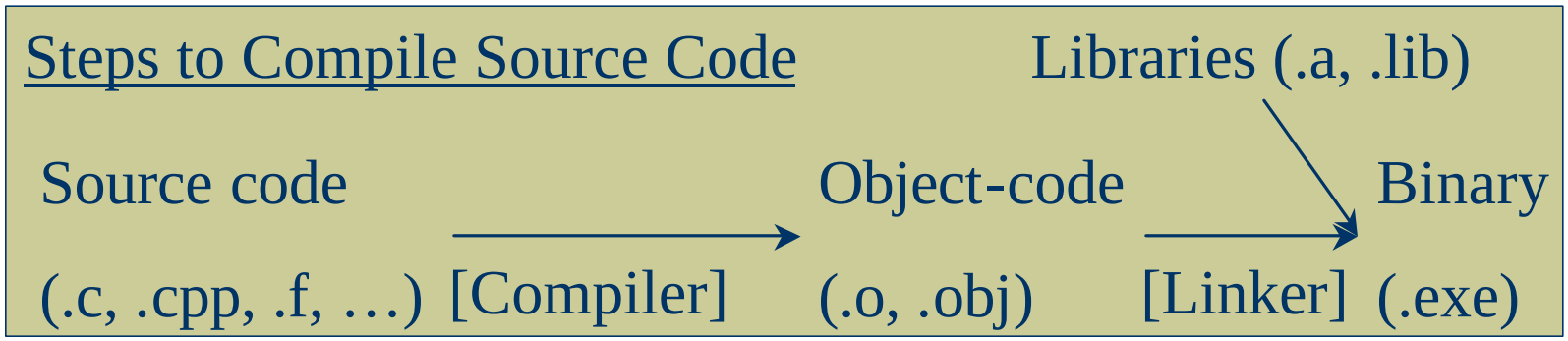
- Port: Sun Solaris to Win32
- Rocket: Like SPP, lower fidelity, faster
- DATCOM: Aircraft and missile stability and control characteristics predictions
- Why Port: Unix workstation harder to come by than PCs

Rocket and DATCOM Ports: Tools Used

- Cygwin – Unix layer on top of Win32
- XFree86 – Widely used X server
- Lesstif – Motif-compatible library
- GCC – GNU Compiler Collection (C and Fortran used)

Rocket and DATCOM Ports: Current Status

- Rocket: Port completed
  - Already in use by Dynetics and our govt sponsor
- DATCOM: Port 75% complete
  - Most C code ported
  - Still to go: C and Fortran object-code linking



Outline

- Geometry Tester
- Rocket and DATCOM Ports
- *JMASS, Joysticks, and Simulation Viewers, Oh My!*

JMASS UAV Simulations: Runtime User Input and Simulation Viewing

- Joystick
- Jmass-vIewer Link (JIL)
- Joystick and JIL: The Big Picture
- Demonstration

Joystick

- Goals
  - Human interface to send data into JMASS simulations
  - Platform-independent API
  - Work around having to include “windows.h” directly into JMASS code

Joystick: Continued

- Development Process
  - Wrote simple application that read joystick state
  - Developed api
  - Wrote class and test client implementations
  - Integrated with a JMASS simulation
- Used Now
  - Shadow 200 UAV simulation
  - Could be used to do anything that requires user input: radar or tank control, non-JMASS work, etc

Joystick: Future Work

- Add capability in backend for additional platforms (eg X)
- Add sockets option to allow for remote joystick usage

Jmass-vIewer Link (JIL)

- Goals:
  - Allow the viewing of simulations as they are simulated (soft-realtime)
  - Remote viewing (send data over network)
  - Take advantage of already-developed rendering software
  - Easily expanded communications capabilities

Jmass-vIewer Link: Development Process

- Discussed what was needed with simulation and viewer sides
- Developed Interface Control Document
- Wrote the JIL server implementation to be used in the viewer
- Wrote an example client to test the server (now used for regression testing)
- Worked with simulation side to develop a full JIL client inside of JMASS
- System testing

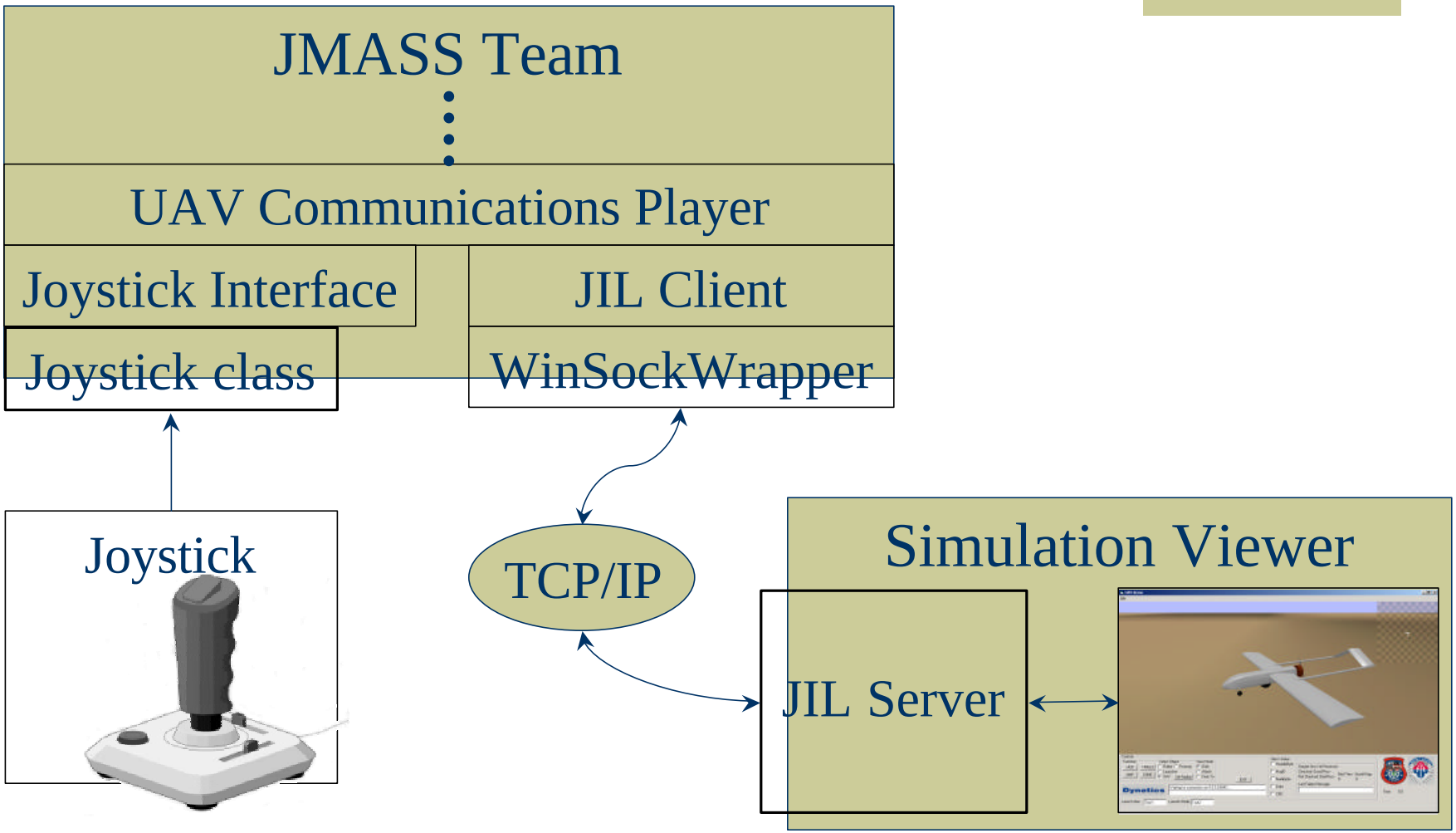
Jmass-vIewer Link: A Typical Message

- Header Byte
- MessageID (Init, data feed, launch, acknowledgement,…)
- Number of Bytes in the Message
- Data
- Checksum

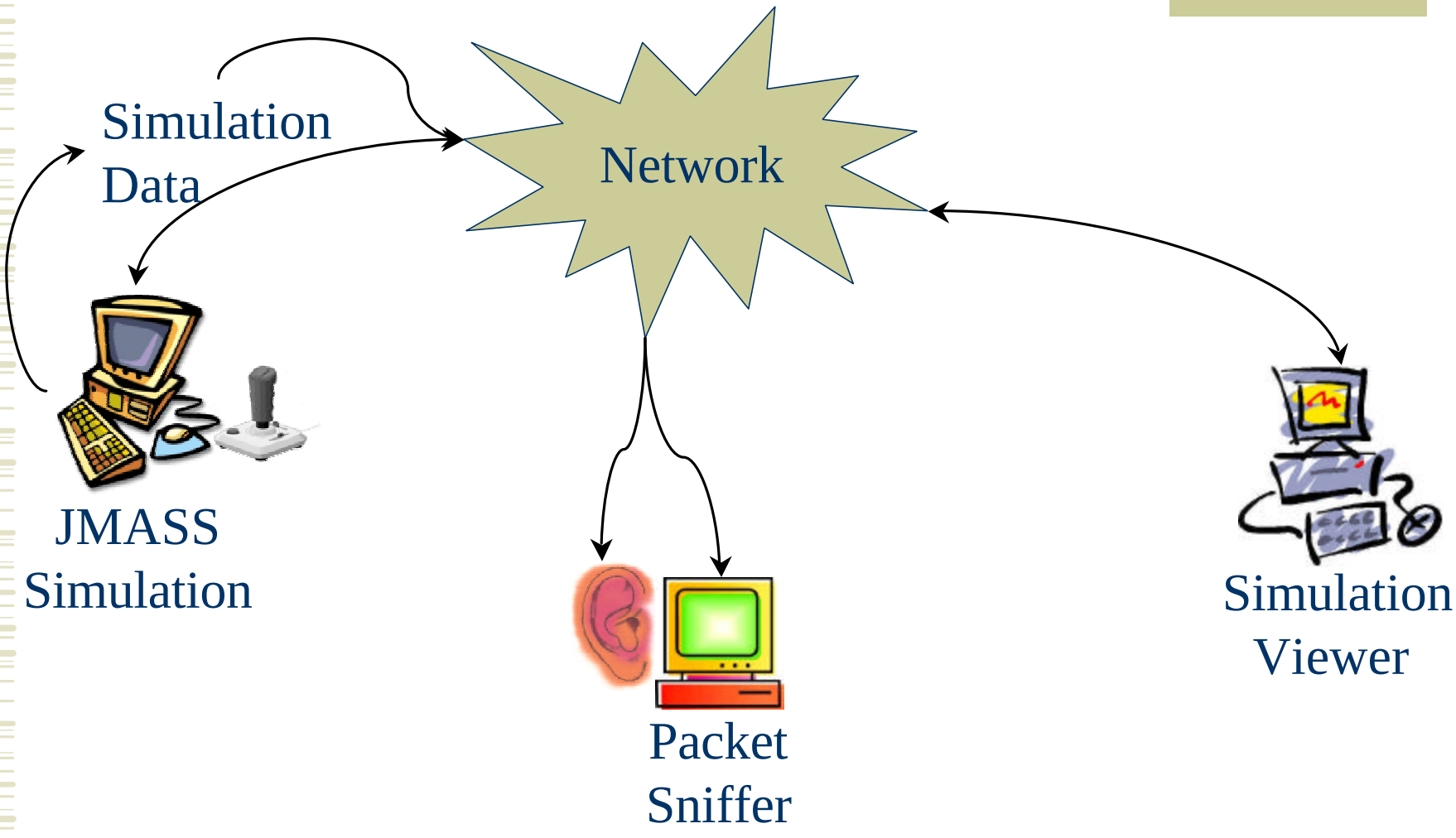
Jmass-vIewer Link: MessageID 1 Data

- Time
- Roll, Pitch, Yaw
- Position (3D rectangular)
- Altitude
- Airspeed

Joystick and JIL: The Big Picture



Joystick and JIL: Demonstration



Lessons Learned

- Communications using sockets
- Using VB at a fairly low level
- Working with compilers/debuggers/linkers
- Using PCP in the workplace
- UAVs
- Solid Rocket Propellants
- Third-party software: a double edged sword
- Classes (Digital Logic Design and Linear Algebra)
- Working in a distributed team
- How to serve a volleyball

Play Time!