The Plan

- Talks
  - Me
  - Biefeld & Lebiere
  - Ball & Gluck
  - Schoelles
  - Bothell
  - Salvucci & Lee
  - Ritter & St. Amant
- General discussion and Q&A with audience

Flying X-Plane with ACT-R

Michael D. Byrne
Department of Psychology
Rice University
Houston, TX
byrne@acm.org
http://chil.rice.edu/
**Background Project**

- NASA grant in 2001 to model pilot taxiing a 737(-ish) plane through O’Hare
  - Were not given the ability to interact with the same simulator code as used in the experiment
  - Did get the database of all the polygons, though
- Pretty much re-implemented everything in Lisp
  - Aircraft braking & acceleration model based on scaled-up version of Salvucci’s code
  - Visual environment based on the NASA database
- Important lessons
  - High-fidelity aircraft model critical
  - A pain to do without a jock Lisper around

**New NASA Work**

- Simulate a pilot during final approach and landing
  - Once again, not given experiment code
  - Significantly more complex dynamics than just accelerating and braking
    - Fidelity to real 757 important to timing
- Our approach
  - Mock up relevant cockpit displays in Lisp
  - Hook up to a separate flight simulator
    - Must have accurate flight dynamics and an autopilot
    - Must be able to talk to it in both directions
  - Our choice: X-Plane (www.x-plane.com)
    - Approved by FAA for real-world pilot training
The Setup

“Device” built in MCL

UDP communication

Aircraft state
Pilot actions

Lisp function calls
(standard RPM stuff)

ACT-R 5

X-Plane

X-Plane
What is Communicated

- ACT-R productions do their usual thing
  - Look around the screen
  - Commands issued by speech to virtual PNF

- Big issue is the Display $\leftarrow\rightarrow$ X-Plane link
  - X-Plane is designed for network “play,” e.g. multiple planes simultaneously in the same airspace
  - X-Plane sends out things like altitude, airspeed, etc. over network interface
  - Display code has routines which render the display based on these values
  - Limited communication from GUI to X-Plane
    + O.K. for this task, which is mostly monitoring the autopilot

How Is Communication Achieved

- UDP Network Interface
  - UDP lives below/beneath the TCP/IP socket level, i.e., **very** primitive
    + No guarantee/verification of packets or anything
    + Very raw data format
  - However, not uncommon for games to use UDP

- X-Plane side
  - Sends out state information at 20Hz

- GUI side
  - Attempts to pick up network packets at ~15 Hz
  - Translates strings spoken by ACT-R to appropriate commands (limited vocabulary)
**Issues**

- Do not try to update the MCL-based display too often!
  - UDP packets sometimes vanish for no apparent reason
    - Buffer all state and only update the buffer for packets that you actually get
    - Run display updates from the buffer, not from network
  - It’s slow and hurts real-time performance
  - ACT-R doesn’t need it anyway
    - Doesn’t access the display all that fast
    - Cuts down on vision auto-update problems
- X-Plane has its own bugs and problems
  - Documentation, hand-configuration, etc.

**Good News and Work in Progress**

- **Success**
  - Overall works pretty well
  - Nice to be able to run flight sim on separate machine from ACT
    - Could probably even run it on a <gasp> PC
- **In progress:** Rendering something like what the pilot can see out the window
  - No information sent by X-Plane about what can be seen out the window
  - Will have to work it out based on location of airplane, which is sent
  - Fortunately, real pilots don’t do all that much of this
X-Plane
- Runs in real time
- Doesn’t have facilities for time synchronization

Therefore, we run ACT-R in “real time” mode and rely on this to keep everything synchronized

No problems with this scheme so far
- Except that it can’t run faster than real time

Acknowledgments
- Dave Huss at Rice
- Rei-sung Lin at UIUC