Icing Encounter Flight Simulator

Principal Investigator: Michael Selig

Graduate Students: Bipin Sehgal, Robert Deters*

Undergraduate Students: Michael Savchenko†

* Started January 2001 † From March 2001 – May 2001
SMART ICING SYSTEMS Research Organization

Core Technologies

- Aerodynamics and Propulsion
- Flight Mechanics
- Control and Sensor Integration
- Human Factors
- Aircraft Icing Technology

IMS Functions

- Characterize Icing Effects
- Operate and Monitor IPS
- Envelope Protection
- Adaptive Control

System Integration

- Flight Simulation
- Flight Test
Icing Encounter Flight Simulator

**Goal:**
- Improve the safety of aircraft in icing conditions.

**Objectives:**
- Function as a *systems integrator* by bringing together the various flight simulator components composed of an aircraft model, flight mechanics, aerodynamics, propulsion, controls, sensors, the ice protection system, the smart icing system, and human factors
- Perform "virtual flight tests" to examine the effects of icing on aircraft operations under a variety of conditions

**Approach:**
- Develop an Icing Encounter Flight Simulator
- Apply the simulator to icing scenarios and experiments
Outline

Bipin Sehgal: Project Overview
Flight Gear Flight Simulator (FGFS)
UIUC-FGFS
- Code Organization
- SIS Components Integration
Graphics

Robert Deters: UIUC-FGFS
- Aircraft Model
- Icing Model
Simulator Usage

Michael Selig: Demonstration
Conclusion
Progress as of Previous Review

- Improvement to the aerodynamics model
- Integration of SIS components
  - Icing model
  - Parameter ID algorithm
  - Hinge moments
  - Icing characterization neural network
- Improved graphics and features
SMART ICING SYSTEMS Research

Aerodynamics and Propulsion
Flight Mechanics
Control and Sensor Integration
Human Factors
Aircraft Icing Technology

IMS

Aerodynamics Model
Icing Model
Icing Encounter Flight Simulator
Simulator Design
Graphics
SIS Components Integration

THE ICING ENCOUNTER FLIGHT SIMULATOR GROUP
Flight Gear Flight Sim (FGFS)

- Project started July 1997 (led by Curt Olson of Univ. of Minnesota Human Factors Lab)
- Cooperative development, over 50 people have contributed (http://www.flightgear.org)
- Free code, open-source, C/C++, about 100,000 lines of code, adheres to GNU General Public License (GPL)
- OpenGL graphics
- Multi-platform:
  - Windows NT/2000 (i86 platform)
  - Windows 95/98
  - Linux (any platform)
  - BSD Unix
  - SGI IRIX
  - SunOS
  - Macintosh
Flight Gear Capabilities

- Pilot input options:
  - Keyboard
  - Mouse (operates as joystick)
  - Joystick, throttle, rudder pedals

- Flight dynamics model:
  - Uses NASA Langley LaRCsim
    (Bruce Jackson, NASA TM 110164, Apr. 1995)
  - Models full 6-DOF nonlinear equations of motion
  - Uses quaternions for coordinate transformation
Flight Gear Capabilities

- Aircraft models:
  - Navion
  - Cessna 172
  - Cherokee
- Autopilots:
  - Altitude hold
  - Heading hold
  - Configured for Navion aircraft model only
- Views:
  - Instrument panel add-on
  - Realistic out-of-cockpit terrain view with HUD
  - Additional world scenery downloads available
Downloadable Scenery
Sample Screen Grab – On Ground
Flight Gear Graphics

Mountains and Deserts
Cities, glaciers, bodies of water, Sun
Flight Gear Graphics

Actual vs. simulated terrain
Flight Gear Graphics

Fog and clouds
Flight Gear Graphics

Night view
Sample Screen Grab - In Flight
Recent Scenery Improvements
Three Panel Flight Gear Setup
Flight Gear Requirements

- Compiling tools:
  - Mesa (OpenGL clone)
  - Portable game library (plib)
  - Automake and autoconf (require perl)
  - Windows 95/98/NT/2000
    - Cygwin (UNIX shell environment for windows)
    - MSVC++
  - Linux (Redhat 7.1)
    - MATLAB 6.0 compiler and math library (for neural network code)
Flight Gear Requirements

• Development tools:
  – Source Navigator
  – Linux, Windows NT 4.0 (Cygwin 1.3)
  – GNU C++ compiler
  – MATLAB 6.0 for Linux
    • MATLAB compiler
    • MATLAB C++ math library
UIUC-FGFS

• Platforms:
  – Windows NT
  – Linux (Redhat 7.1)

• Based on Flight Gear 0.7.6

• Current capabilities:
  – Reconfigurable aircraft flight model
  – User can load aircraft-specific data at runtime using keyword-based input file
    • Linear stability derivatives
    • Nonlinear tabulated data (linear interpolation)
About 20 aircraft models available, including:

- Twin Otter (a)
- Beech 99
- Pioneer UAV (b)
- Cessna 172
- Marchetti S-211
- Learjet 24
- Convair 880
- Boeing 747 (c)
- F-4 Phantom II
- F-104 Starfighter (d)
- X-15 (d)
UIUC-FGFS

- Icing characterization (linear $k_{C_\eta_{ice}}$ model)
- Flight data recorder
- Networked instrument views building on Brian Fuesz’s (Frasca) WinPioneer code
- Multiple aerodynamic model formats accepted:
  - Standard linear model (Roskam)
  - FDC (Twin Otter) linear model (Rauw)
  - Pioneer nonlinear model (Bray)
Sample Input File Commands

init          recordRate      10      # [times/s]
geometry     cbar            6.50    # [ft]
controlSurface de          20 20    # [deg]
mass         Weight          11000   # [lb]
mass         I_xx           16039   # [slug-ft^2]
engine       simpleSingle    2000    # [lb]
CD            CDo            0.360   # [-]
CD            CXo            -0.360  # [-]
CL            CZ_de          -0.608   # [1/rad]
CL            CLfade CLfade.dat 0 1 1 # [-,deg,deg]
Cl            Cl_dr          0.015    # [1/rad]
record       Simtime         0.015    # [s]
record       Altitude      # [ft]
record       Alpha_deg     # [deg]
controlSurface elevator_doublet 10 240 2
UIUC Aircraft Model

**uiuc_aerodeflections()**
Determine control surface deflections

**uiuc_coefficients()**
Sum aerodynamic coefficients

**uiuc_ice()**
Calculate iced coefficients

**uiuc_menu()**
Read input file & initialize variables

**uiuc_wrapper()**
Calculate aircraft forces & moments

**uiuc_aero()**
Pass parameters to **uiuc_wrapper()**

**LaRCsim**
Compute new aircraft state

**uiuc_recorder()**
Output data

$t > 0$
$t = 0$
$t > t_{ice}$
SMART ICING SYSTEMS Research

Smart Icing Systems Review, June 19-20, 2001

Aerodynamics and Propulsion

Flight Mechanics

Control and Sensor Integration

Human Factors

Aircraft Icing Technology

Aerodynamics Model

Simulator Design

Graphics

Icing Model

Icing Encounter Flight Simulator

IMS

SIS Components Integration

THE Icing ENCOUNTER Flight SIMULATOR GROUP
SIS Components Integration

- MATLAB code for:
  - Hinge moments
  - Parameter ID algorithm
  - Icing characterization neural network

- Code converted into C++ using MATLAB APIs
  - Approximately 12,000 lines of C++ code
  - Need MATLAB compiler and math libraries for code conversion
  - Compiled using GNU C++ compiler
Neural Network Architecture

Turbulence

UIUC-FGFS

Measurement Noise

Hinge Moments

Excitation Measure

Trim Characterization

Parameter ID Algorithm

Clean Trim Characterization

Icing Characterization Neural Network

MATLAB code conversion to C++ validated ✓
SMART ICING SYSTEMS Research

Smart Icing Systems Review, June 19-20, 2001

IMSI

Aerodynamics and Propulsion
Flight Mechanics
Control and Sensor Integration
Human Factors
Aircraft Icing Technology

Aerodynamics Model
Simulator Design
Graphics
Icing Model
Icing Encounter Flight Simulator
SIS Components Integration

Robert Deters
Bipin Sehgal
Michael Savchenko
Prof. Selig

THE ICING ENCOUNTER FLIGHT SIMULATOR GROUP
Graphics

- Switch to Flight Gear 0.7.6 (better graphics)
- Cloud appearance as a function of time (with icing)
- UIUC instrument panel
  - Human factors
    - What do we display to the pilots?
    - Design the IMS flight-deck display concepts
  - Implementation into OpenGL code
  - Integration with UIUC-FGFS
Instrument Panel

Flight Gear instrument panel
Glass Cockpit
New Glass Cockpit
SIS Glass Cockpit

- Based on Brian Fuesz’s implementation in WinPioneer
- OpenGL implementation under Linux
- Over the network on a different PC
- Ice detection based on $\bar{\eta}$ generated by the neural networks
Summary

- UIUC code adopted by Flight Gear group
- Basic aerodynamics model now functional
- Easy to add new aircraft models
  - Some 20 models already available
- Easy to expand code for new variables
- Ability to detect the onset of icing using neural networks
SMART ICING SYSTEMS Research

Aerodynamics and Propulsion
Flight Mechanics
Control and Sensor Integration
Human Factors
Aircraft Icing Technology

Aerodynamics Model
Icing Model
Icing Encounter Flight Simulator

Simulator Design
Graphics
SIS Components Integration

Robert Deters
Bipin Sehgal
Michael Savchenko
Prof. Selig

THE ICING ENCOUNTER FLIGHT SIMULATOR GROUP
Outline

• Background Flight Dynamics
  – Code Layout
  – LaRCsim Module

• UIUC Aeromodel
  – Architecture
  – Icing Model
  – Gear Model
  – Validation Required Additions

• Summary

• Future Research
Flight Dynamics Module

Flight Gear

FDM

JSBsim

LaRCsim

UIUC-Aeromodel
LaRCsim Module

- LaRCsim flight dynamics model
  - Aircraft models:
    - Navion
    - Cessna 172
    - Piper Cherokee
  - Equations of motion:
    - Aerodynamic forces computed in the wind axis system:
      \[
      \begin{align*}
      F_{x,wind} &= -qSC_D \\
      F_{y,wind} &= qSC_y \\
      F_{z,wind} &= -qSC_L
      \end{align*}
      \]
LaRCsim Module

– Wind-axis to body-axis transformation gives the body axis forces:

\[
\begin{bmatrix}
F_{xaero} \\
F_{yaero} \\
F_{zaero}
\end{bmatrix} =
\begin{bmatrix}
\cos \alpha & \cos \beta & -\cos \alpha \sin \beta & -\sin \alpha \\
\sin \beta & \cos \beta & 0 \\
\sin \alpha \cos \beta & -\sin \alpha \sin \beta & \cos \alpha
\end{bmatrix}
\begin{bmatrix}
F_{xwind} \\
F_{ywind} \\
F_{zwind}
\end{bmatrix}
\]
LaRCsim Module

– Aerodynamic moments computed in body axis system:

\[ M_{\text{body}} = qSbC_l \]
\[ M_{m\text{body}} = qS\bar{c}C_m \]
\[ M_{n\text{body}} = qSbC_n \]

– Aerodynamic forces and moments summed with others acting on the aircraft:

\[ F = F_{\text{aero}} + F_{\text{engine}} + F_{\text{gear}} + \ldots \]
\[ M = M_{\text{aero}} + M_{\text{engine}} + M_{\text{gear}} + \ldots \]
LaRCsim Module

- New aircraft state (i.e. roll, pitch, yaw rates, angle orientation etc.) determined
- LaRCsim adapted for UIUC aerodynamics model
UIUC Aeromodel Architecture

uiuc_menu()
Read input file & initialize variables

uiuc_coefficients()
Sum aerodynamic coefficients

uiuc_engine()
Engine forces & moments

uiuc_gear()
Landing gear forces & moments

uiuc_ice()
Compute iced coefficients

uiuc_aerodeflections()
Determine control surface deflections

uiuc_recorder()
Output data

uiuc_wrapper()
Calculate aircraft forces & moments

LaRCsim
Compute new aircraft state

\[ t = 0 \]
\[ t > 0 \]
\[ t > t_{ice} \]
Icing Aerodynamics Model

- Applied $\eta_{\text{ice}}$ model developed by Prof. Bragg's group
  - Twin Otter model

\[
C_{(A)_{\text{iced}}} = \left(1 + \eta_{\text{ice}} k'_{C_{(A)}} \right) C_{(A)}
\]
Icing Aerodynamics Model

• Allows $\eta_{\text{ice}}$ and icing constants ($k_{C_\text{c}}$) to be varied to model different icing cases
  – Wing icing
  – Tail icing
  – Aircraft icing
  – Overall icing severity
Sample of Different Icing Cases

<table>
<thead>
<tr>
<th></th>
<th>All Ice</th>
<th>Wing Ice</th>
<th>Tail Ice</th>
</tr>
</thead>
<tbody>
<tr>
<td>kCXo</td>
<td>6.52696</td>
<td>2.64444</td>
<td>1.58844</td>
</tr>
<tr>
<td>kCX_a</td>
<td>-0.14296</td>
<td>-0.03156</td>
<td>-0.04504</td>
</tr>
<tr>
<td>kCZ_a</td>
<td>-1.48148</td>
<td>-0.83259</td>
<td>-0.36593</td>
</tr>
<tr>
<td>kCZ_de</td>
<td>-1.40741</td>
<td>-0.33970</td>
<td>-1.05556</td>
</tr>
<tr>
<td>kCm_a</td>
<td>-1.46667</td>
<td>-0.28346</td>
<td>-0.53244</td>
</tr>
<tr>
<td>kCm_de</td>
<td>-1.48148</td>
<td>-0.26504</td>
<td>-1.24756</td>
</tr>
</tbody>
</table>
Gear Model

- Created by David Megginson (FGFS) based on Tony Peden’s (FGFS) work
- Configure landing gear by commands in the input file
  - Location
  - Damping constant
  - Spring constant
  - Rolling friction coefficient
Aerodynamic Model Additions

- Features added for flight test validation task
- Ability to start at any initial condition
  - FGFS command line inputs
    - Altitude, velocity (u,v, w), Euler angles (φ, θ, ψ)
  - UIUC input file commands
    - Angular velocities (p, q, r), control surface deflections (aileron, elevator, rudder), throttle, angle of attack, sideslip angle
Aerodynamic Model Additions

- Ability to run pre-made flight maneuvers
  - Basic maneuvers superimposed on stick flying
    - Elevator step, singlet, and doublet
  - Full flight test / complex maneuvers
    - Elevator, aileron, rudder, and throttle input files
      - Function of time
      - Linear interpolation between points
Simulator Usage

- Running the simulator
  - Need a good 3D accelerator card with full OpenGL drivers to achieve smooth frame rates
  - Runs through the command line or by using a batch file
Simulator Usage

rem @ECHO OFF

rem Skip ahead to CONT1 if FG_ROOT has a value
IF NOT %FG_ROOT%==. GOTO CONT1

SET FG_ROOT=.

:CONT1

rem Check for the existence of the executable
IF NOT EXIST %FG_ROOT%\BIN\FGFS.EXE GOTO ERROR1

rem Now that FG_ROOT has been set, run the program
ECHO FG_ROOT = %FG_ROOT%

rem Twin Otter at Chicago (O'Hare Airport)
%FG_ROOT%\BIN\FGFS.EXE --aircraft-dir=Aircraft-uiuc/TwinOtter --airport-id=KORD --heading=45

rem Marchetti trainer at Grand Canyon
rem %FG_ROOT%\BIN\FGFS.EXE --aircraft-dir=Aircraft-uiuc/Marchetti --airport-id=25U --heading=10
--- runfgfs.bat<2> 12:45PM (Text Fill)--L18--C0--Top---
Sample Input File

emacs@AA16

Buffers  Files  Tools  Edit  Search  Mule  Makefile  Help

init recordRate 4  # [/s]
#init recordStartTime 5  # [s]
init Dz_cg 2.5  # [ft]  FG c172_aero.c
init nondim_rate_V_rel_wind

gometry bw 35.8  # [ft]  FG c172_aero.c
gometry cbar 4.90  # [ft]  FG c172_aero.c
gometry Sw 174.0  # [ft^2]  FG c172_aero.c

controlSurface de 28 23  # [deg]  FG c172_aero.c
controlSurface da 20 15  # [deg]  FG c172_aero.c
controlSurface dr 16 16  # [deg]  FG c172_aero.c
controlSurface zero_Long_trim
#controlSurface set_Long_trim_deg 1.6598
#controlSurface elevator_doublet 10 240 2

mass Weight 2300.0  # [lb]  FG c172_init.c
mass I_xx 948.0  # [slug-ft^2]  FG c172_init.c
mass I_yy 1346.0  # [slug-ft^2]  FG c172_init.c
mass I_zz 1967.0  # [slug-ft^2]  FG c172_init.c
mass I_xz 0.0  # [slug-ft^2]  FG c172_init.c

engine c172

CL CL_adot 1.7  # [/rad]  FG c172_aero.c
CL CL_q 3.9  # [/rad]  FG c172_aero.c
CL CL_de 0.34763715  # [/rad]  FG c172_aero.c
CL CLfa ../Aircraft-uiuc/000504-Cessna172-TD2/CLfa.dat 0 1  # []  FG c172_aero.c

CD CDo 0.036  # []  FG c172_aero.c
--u** aircraft.dat 12:27PM (Makefile CVS:1.3)--L29--CO--Top----------
Fontifying aircraft.dat... done
Summary

- Model ready for simulating flight tests
  - User specified initial conditions
  - Pre-made flight maneuvers
  - Control surface and throttle input files
- Began validation with flight test data
- Different icing cases ready for testing
  - Wing icing
  - Tail icing
  - Aircraft icing
  - Icing severity
Future Work

• Code maintenance: keeping up-to-date with the Flight Gear improvements

• Simulator improvements
  – Icing model
  – Engine model
  – Gear model
  – Nonlinear aerodynamics
  – Turbulence
Future Work

• Incorporate SIS components
  – Latest ID algorithm
  – Autopilot
  – Human factors
• Perform virtual flight tests for the Twin Otter to aid in the design of the next set of experiments