BlueSky: Open Source, Open Data Approach to Air Traffic Simulation

Jacco Hoekstra
Outline

• Why open source, open data traffic simulation?
• Open source challenge
• Open data challenge
• Current prototype
• Applications
• Future work and work in progress
• Your contribution
Current status of ATM Science

now \rightarrow A

Feasible!
Better!
now

A

Feasible!
Better!

B

Feasible!
Better!
Feasible!

Better!

Feasible!

Better!

Feasible!

Better!

now

A

B
Feasible!
Better!
Feasible!
Better!
Feasible!
Better!
Research lines CNS/ATM group TU Delft

1. **Foundation of ATM research**
   1. Open source tools (ADS-B decoding, ATM simulation, conversion tools)
   2. Open Data (Performance data, scenarios, historic traffic data)
   3. Big Data & Metrics (Complexity, Traffic Flow dynamics)

2. **ADS-B surveillance and applications**
   1. Raw data analysis, surveillance quality
   2. Airborne Applications (ATSAW, CD&R algorithms)

3. **Drone CNS/ATM technology**
   1. Indoor & outdoor navigation and autonomy
   2. Swarming

4. **General Aviation CNS/ATM technology**
   1. Airborne radar & surveillance technology for safety

5. **Schiphol/Mainport related CNS/ATM issues**

Google: *ADS-B Junzi Sun*

Google: *ProfHoekstra BlueSky*
Improve science & research in ATM

- **Comparable** research results, same metrics, tools and scenarios

- Research does not thrive under **standardisation**

- But every **researcher needs** aircraft performance models => success
  BADA 3

- **Open Source, free** (also for tools)

- **Open Data, free**
Previous work: BADA, “only” aircraft data

- Data files with performance and procedure data
- BADA 3 vs BADA 4
- Quality BADA 3 could be improved easily
- BADA 4 has more sensitive, proprietary data
- BADA 3 currently default standard for research
- Licensing regime tightens
- Not fully open
Previous work: Cassiopeia

- Multi-agent network simulation
- Client-server structure
- JADEX simulation engine
Previous work: ELSA

- Flight Plan inputs
- Similarity with the project
- Conflicts are simulated and analyzed
- Input and output via data files
- Python with C-parts
Previous work: Traffic Manager (TMX)

- NLR Traffic Manager: Fast-time simulation and real-time simulation
- Batch simulation
- Open architecture, but Visual studio Visual C++ with Fortran core
- Many features and proprietary data
- Used by many partners of NLR such as NASA, DLR, TU Delft, universities
BlueSky based on these experiences

• For use in academic world:

• 100 % fully open data:
  • Aircraft performance data
  • Navigation data

• 100 % open source:
  • Co-development shared via GitHub/ProfHoekstra/BlueSky
  • Should not require any commercial development tools or libraries
  • Quality control?

• Increase adoption:
  • Easy to use: GUI and data/scenarios in plain text files, run locally
  • Readable source code for scientists who are not computer experts
  • Philosophy: no need to understand what you do not need to know
BlueSky GNU General Public License v3

• **Required:**
  - Disclose Source
  - License and copyright notice
  - State Changes

• **Permitted:**
  - Commercial Use
  - Distribution
  - Modification
  - Patent Use, Private Use

• **Forbidden:**
  - Hold Liable

Challenges Open Data vs Quality?

- **Navigation data:**
  - Geographical information
  - Navigational Aids
  - Waypoints
  - Airport data: taxiways, runways
  - Sector lay-out/Airspaces

- **Aircraft Performance data:**
  - Drag polar
  - Engine performance
  - Operating weights
  - Autopilot/Autothrottle settings, mode logic
  - Procedural speeds

- **Weather data?**
Open data: navaids

- Web-crawling programs collect the data from public sites and convert to text data files
- Global coverage for navaids and waypoints
- No guarantee that it is always up-to-date
Open Data: Geographical information

• Just background for GUI

• Geo-website contains all required info:
  • Coastlines
  • Rivers
  • Borders

• Satellite image used from Google Earth
Open data: Airports

- 14480 airports included
- Large airports also feature taxiway and runway layout
- Flight simulator and gaming community have people who collect this information as well
Open Data: FIRs, Sectors etc

- Still very limited
- Web-crawling possible?
- Now implemented manually based on scenarios
- Console is not ATCo HMI
- More help needed
Open Data: Aircraft Performance models

- Compatible Plug & Play with BADA v3.12 (nearly open): Copy BADA files to empty folder: `data\coefficients\BADA`
- Generic Open Data models fully based on Open sources (built-in)
- Big Data effort using ADS-B data to develop comprehensive set of aircraft performance files (built-in)
- Metrics also use generic measure Energy, independent of Fuel Flow
Open Source: choice for platform

- BlueSky simulator made entirely in Python
- Based on standard scientific libraries as defined in Python(x,y) (and Anaconda)
- Python 2.x
- Uses Numpy, Scipy
- For GUIs:
  - Qt, OpenGL (included)
  - and/or Pygame SDL
Open Source: choice for platform

- BlueSky simulator made entirely in Python

- Based on standard scientific libraries as defined in Python(x,y) (and Anaconda)

- Python 2.x

- Uses Numpy, Scipy

- For GUIs:
  - Qt, OpenGL (included)
  - and/or Pygame SDL

- No need to understand this!
Open Source: choice for platform

- BlueSky simulator made entirely in Python
- Based on standard scientific libraries as defined in Python(x,y) (and Anaconda)
- Python 2.x
- Uses Numpy, Scipy
- For GUIs:
  - Qt, OpenGL (included)
  - and/or Pygame SDL
- No need to understand this!

```python
from math import *

print "Hello ICRAT"

a,b,c = input("Give a,b,c":)

D = b**2-4.0*a*c

top = -b/(2.0*a)

if D>=0.0 :
    x1 = (-b-sqrt(D))/(2.0*a)
    x2 = (-b+sqrt(D))/(2.0*a)
    print "x = ",x1," or x =",x2
else:
    print "No solutions"
```

Example of readability of Python source code
if wpok:
    # Overwrite existing origin
    if self.nwp > 0 and self.wptype[0] == self.orig:
        self.wpname[0] = name.upper()
        self.wptype[0] = wptype
        self.wpplat[0] = wpplat
        self.wplon[0] = wplon
        self.wpalt[0] = alt
        self.wpspd[0] = spd
        self.wpflyby[0] = self.swflyby

    else:
        self.wpname = [name.upper()] + self.wpname
        self.wptype = [wptype] + self.wptype
        self.wpplat = [wpplat] + self.wpplat
        self.wplon = [wplon] + self.wplon
        self.wpalt = [alt] + self.wpalt
        self.wpspd = [spd] + self.wpspd
        self.wpflyby = [self.swflyby] + self.wpflyby

        self.nwp = self.nwp + 1
        if self.iactwp > 0:
            self.iactwp = self.iactwp + 1

    idx = 0

    # DESTINATION: Wptype is destination?
    elif wptype == self.dest:

        if not (name == traf.id[iac] + "DEST"):
            # published identifier
            i = self.navdb.getapidx(name.upper(), strip())
            if wpok:
                wpplat = self.navdb.aplat[i]
                wplon = self.navdb.aplon[i]
                # lat/lon type
            else:
                wpplat = lat
                wplon = lon
                wpok = True

        # Overwrite existing destination
Modules inside BlueSky

BlueSky Main program

Simulation Engine

- Command Stack
- Scenario File reading

Command Processing

Traffic simulation

- Traffic Performance
- FMS Routes
- Autopilot and AutoThr
- Traffic Dynamics
- Conflict Detection
- Conflict Resolution

Navigation Database

Datafeed interface

Network interface

Metrics

- ADS-B Antenna feed

- Graphics Set-up
- Radar Screen
- Edit/Console Window

Screen Module

User interface

- Keyboard & Commands
- Mouse cmd completion
GUI: Classic (Pygame/SDL based)
GUI: Advanced (Qt/OpenGL based)
Download & set-up
GitHub>ProfHoekstra>BlueSky

• Easiest way to set up BlueSky:
  • Download & install full Python(x,y) suite http://python-xy.github.io/
  • Download & install pygame (win32 version for Python 2.7): http://pygame.org/ftp/pygame-1.9.2a0.win32-py2.7.msi

• Now you have the Python environment, then download and install BlueSky, Choose Clone or Download, then Download as Zip:
BlueSky setting up

- Run for BlueSky.py

- The first time (when no settings.cfg exists), answer question: Try 1
Choose yourself which user interface you want

- Delete settings.cfg and run BlueSky.py again

- Or double click on (create a shortcut to)
Scenario file

- Scenario file are basically time stamped input by a user

<table>
<thead>
<tr>
<th>Directory</th>
<th>Date/Time</th>
<th>Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>bluesky</td>
<td>04-Apr-16 08:08</td>
<td>File folder</td>
</tr>
<tr>
<td>data</td>
<td>20-May-16 14:15</td>
<td>File folder</td>
</tr>
<tr>
<td>extra</td>
<td>30-May-16 15:22</td>
<td>File folder</td>
</tr>
<tr>
<td>info</td>
<td>22-Mar-16 11:35</td>
<td>File folder</td>
</tr>
<tr>
<td>scenario</td>
<td>19-May-16 10:54</td>
<td>File folder</td>
</tr>
<tr>
<td></td>
<td>19-Mar-16 07:03</td>
<td>File folder</td>
</tr>
<tr>
<td>utils</td>
<td>04-Apr-16 08:07</td>
<td>Text Document</td>
</tr>
<tr>
<td></td>
<td>07-Apr-16 10:52</td>
<td>Python File</td>
</tr>
<tr>
<td></td>
<td>07-Apr-16 10:52</td>
<td>Python File</td>
</tr>
<tr>
<td></td>
<td>19-May-16 10:54</td>
<td>Python File</td>
</tr>
<tr>
<td></td>
<td>19-May-16 16:50</td>
<td>Compiled Python File</td>
</tr>
<tr>
<td></td>
<td>19-Mar-16 07:02</td>
<td>Python File</td>
</tr>
<tr>
<td></td>
<td>21-Apr-16 11:10</td>
<td>MD File</td>
</tr>
<tr>
<td></td>
<td>18-Jun-16 11:32</td>
<td>CFG File</td>
</tr>
</tbody>
</table>
Scenario file

- Scenario file are basically time stamped input by a user
Scenario file: demo.scn
Run BlueSky:

- In QtGl version: Enter “IC” to select a file or “IC demo” to start demo file (“IC de”+Tab uses file completion) as Initial Condition

- In pygame version: select demo.scn
Language in console and files: TrafScript

- CRE ?
  CRE acid,type,lat,lon,hdg,alt,spd
- CRE KL204,B747,52,4,090,FL100,250
Language in console and files: TrafScript

- CRE?
  CRE acid,type,lat,lon,hdg,alt,spd
- CRE KL204,B747,52,4,090,FL100,250

Click on an aircraft (once to select, double to see status)
- KL204 HDG 270  (call sign and command can be reversed)
- ALT KL204, FL250

- PAN KJFK and zoom in with zoom buttons or gestures on touchscreen
- QUIT stops program
- IC allows selecting a new scenario file

- ADDWPT?
  KL204 DEST EHAM  Flight plan entered like an FMS
  KL204 ORIG KJFK  Also see commands LNAV, VNAV, DIRECT, etc.
  ADDWPT acid, (wpname,lat,lon),[alt],[spd],[afterwp]
- KL204 addwpt SPY,FL240,350
TrafScript: ease of use, compatibility

- **Easy** to use in *console* or in *time-stamped scenario file*

- Scenario files are **text files** can be edited in Notepad or Word

- Scenarios **compatible** with tools like NLR Traffic Manager TMX or NASA LaRC ATOL lab

- **Converters** for Eurocontrol DDR2 are in beta phase, other converter can easily be added due to simplicity TrafScript language

- **Scenario generation tools** run external

- Requires only **basic computer skills** from user
BlueSky already contains many features

- FMS and autopilot logic
- ASAS/ATC simulation for conflict detection and resolution (different modules)
- Extensive scenario control
- External connections
- Batch simulation on multiple CPUs and multiple PCs, very fast, thousands of aircraft (QtGL version only)
- Metrics & data logging
- Visualisation: traffic samples, ADS-B plug-in
Applications so far

- **Complexity metrics** in existing traffic demand scenarios

- Simulate and investigate effects of ATM procedures like **Upstream delay absorption**

- **AMAN and XMAN** interference of scheduling with pop-up traffic

- **Visualize ADS-B** data from receiver

- **Conflict Detection and Resolution** studies

- **ASAS/Swarming** algorithms

- Effect of procedures and **airspace structure** on capacity and safety
Your help is needed

• Fully open source, so help us:

• **Missing** data in GUI/model: sectors, airspaces, weather/wind data

• **Debugging**: send reports

• Extension: **request us to add functionality** you need

• Extension: add your own **functionality in a fork**

• Wikipedia philosophy: open source and forever beta

• Documentation:
  [https://github.com/ProfHoekstra/bluesky/tree/master/info](https://github.com/ProfHoekstra/bluesky/tree/master/info)
  [http://homepage.tudelft.nl/7p97s/BlueSky/](http://homepage.tudelft.nl/7p97s/BlueSky/)
The End Goal of BlueSky

- A fully open source, open data, with extensive features but still easy to use and develop by everyone without any restrictions or licensing

- Exchange metrics (common measures)

- Exchange scenarios (reference set will be developed in AHMED project)

- Stand on the shoulders of your fellow researchers (giants?)

- Join the BlueSky community

- Experimental.....