JDRAGON/JOSCAR: TOOLS FOR MANEUVER STRATEGY COMPUTATION DEVELOPED IN JAVA AND USING PATRIUS
6th International Conference on Astrodynamics Tools and Techniques (ICATT)
Darmstadt, 17/03/2016

Sumelzo Martinez I., Labourdette P.
Orbital Maneuvers Office (DCT/SB/MO)
Centre National d’Études Spatiales, French Space Agency
INTRODUCTION

DEVELOPMENT ENVIRONMENT
- PATRIUS
- GENIUS

JPSIMU: THE NUMERICAL PROPAGATOR

JDRAGON: COMPUTING MANEUVERS
- PROBLEM STATEMENT
- SOLUTION APPROACH
- DEMO

JOSCAR: END-TO-END SIMULATIONS

CONCLUSIONS
INTRODUCTION

● DRAGON/OSCAR Fortran version [1]:

❖ ATV program (1997-2014)

» ATV phasing and rendezvous maneuvers strategy towards ISS (International Space Station)

❖ GALILEO missions (2011-today)

» Operational design of LEOP (Launch and Early Orbit Phase), phasing and rendezvous scenarios

» Compliant with quality and operational constraints → directly integrated into the Flight Dynamics Centre

● Decision to use JAVA technology → SIRIUS project [2]: basic software layout (PATRIUS) and operational tools (FDS) already started.

● JDRAGON/JOSCAR: not a simple porting from Fortran to JAVA, but redefinition of certain functionalities!
INTRODUCTION

- **JOSCAR**
  - ~ 1000 calls
  - Monte-Carlo analysis tool, permits to test mission plan robustness, simulating variables dispersions that could be experienced along the flight

- **JDRAGON**
  - ~ 10 calls
  - Maneuver strategy computation tool, optimizes a fixed number of maneuvers to reach a non-cooperative target, within a fixed phasing duration and fulfilling certain relative conditions

- **JPSIMU**
  - Numerical propagator, based on the PSIMU Fortran version
DEVELOPMENT ENVIRONMENT

JPSIMU- JDRAGON- JOSCAR

Flight dynamics, Mathematics

GUI

PATRIUS

GENIUS

SIRIUS project workbench

Checkstyle, FindBugs, PMD

We can easily generate an executable (runnable file JAR)

Code development

Checkstyle, FindBugs, PMD

JPSIMU.jar

JDRAGON.jar

We can easily generate an executable (runnable file JAR)
DEVELOPMENT ENVIRONMENT

PATRIUS [3]
(PATrimoine de base de siRIUS)

- **Mathematics**: Dispensions, Matrices, Rotations, Interpolations, Geometry, Numerical Integration
- **Flight dynamics**: Time, Orbits, Frames, Celestial Bodies, SpacecraftState
- **Attitude**: Attitude Laws, Kinematics, Transformations, Guidance Commands, Slew
- **Orbit Determination**: Propagation, Physical Models, Measure and Filtering
- **Spacecraft**: Assembly, Sensors, Mass and Forces
- **Mission**: Events Detection, Maneuvers, Postprocessing

GENIUS
(Generation of Interface for Users of Scientific S/W)

- Units management
- Performing conditional display
- Simplified approach, in particular about events, management (setting actions before/after a certain event is reached)
- Read/write for files directly integrated
- Process management compatible in all OS (thanks to JAVA)
Extrapolate the orbit around the Earth

- Taking into account different forces:
  - Earth potential
  - Third body perturbations
  - Aerodynamic forces
  - Solar Radiation Pressure
  - Ocean/Terrestrial tides

- Defining a maneuver sequence
- Defining an attitude sequence
- Using different numerical integrators
- Identifying orbital events

Frames configuration management

<table>
<thead>
<tr>
<th>From</th>
<th>To</th>
<th>Phenomena</th>
<th>Corrections</th>
</tr>
</thead>
<tbody>
<tr>
<td>GCRF</td>
<td>CIRF</td>
<td>Earth rotation axis around Ecliptic pole</td>
<td>Precession, Nutation</td>
</tr>
<tr>
<td>CIRF</td>
<td>TIRF</td>
<td>Diurnal motion</td>
<td>UT1-UTC difference</td>
</tr>
<tr>
<td>TIRF</td>
<td>ITRF</td>
<td>Earth rotation axis wrt Earth’s crust</td>
<td>Tides, libration, S’ effects, EOPs</td>
</tr>
</tbody>
</table>

Event detectors

- To stop propagation
- To switch between attitude laws
- To trigger impulsive maneuvers
- To define the beginning/end of spread maneuvers
- To identify orbital events of interest: ascending/descending nodes, station visibilities, entering/existing eclipse, AOLs, ...
JPSIMU: THE NUMERICAL PROPAGATOR

Utilization modes

- **Subroutine mode** (called from JDRAGON): Containing the pure tool computations

- **Batch mode**: Read a configuration file (".xml") and calls the subroutine mode

- **GUI mode**: Graphical User Interface permits to create scenarios and launch computations (via the batch mode)

- **Output**: +80 variables, ephemeris & detected events files (".txt")

Validation

- **Battery of tests validating ephemeris generation (coverage > 80%)**:
  - **Thematic validation**: 42 test performed, divided in 7 different tests: initial orbit, earth features/frames configuration, vehicle, force models, maneuver scenario, attitude laws and integrator
  - **GUI mode validation**

- **Code’s quality tests**: compliant with CNES coding standard rules
  - FindBugs, Checkstyle
**Problem statement**

Find \( x := \{ \Delta V_j, \varphi_j \}_{j=1}^{N_m} \)

To minimize \( J(x) := \sum_{j=1}^{N_m} |\Delta V_j| \)

With:

- \( \Delta V_j \): Maneuver \( j \) to optimize
- \( \varphi_j \): Maneuver location \( \rightarrow 2\pi (N_{ch}^j - 1) + \alpha_j \)
  \[ \varphi_j := 2\pi (N_{ch}^j - 1) + \alpha_j \]
  \( N_{ch}^j \): orbit number
  \( \alpha_j \): argument of latitude
- \( N_m \): # Maneuvers to optimize
- \( T \): « Real world » propagation
- \( V_{aim} \): Aimed vector

**Constraints**

**C\(_0\)**, the rendezvous condition:
- \( E_{ta}(t_f) + V_{aim} = T(\Delta V_j, \varphi_j, E_{ch}(t_0)) \) \( j = 1, N_m \)

**C\(_1(\varphi_j)\)**, location limitations:
- \( \varphi_j \in (\varphi_j^{min}, \varphi_j^{max}) \)
- \( \varphi_{j+1} - \varphi_j \in [\Delta \varphi_j^{min}, \Delta \varphi_j^{max}] \)

**C\(_2(\Delta V_j)\)**, module limitations:
- \( \Delta V_j \in [\Delta V_j^{min}, \Delta V_j^{max}] \)
- \( R_j, T_j, N_j \in [Val_j^{min}, Val_j^{max}] \)
JDRAGON: COMPUTING MANEUVERS

Propagate target and chaser orbit (without optimizable maneuvers) until rendezvous date \( E_{ch}, E_{ta} \)

Compute deviations \( b = E_{ta} + V_{aim} - E_{ch} \)

Find \( x = \{ \Delta V_j, \varphi_j \}_{j=1,N_m} \)

To minimize \( J(x) = \sum_{j=1}^{N_m} \Delta V_j \)

Complying with:
- \( \{ \varphi_j \}_{j=1,N_m} \in \text{« n-tuples »} \)
- \( C_0, \text{linearized}: A(\varphi_j)X(\Delta V_j) = b \)
- \( C_2(\Delta V_j) \)

\{P2'\} requires solving a linear problem \( AX = b \) iteratively, with \( A \in \mathbb{R}^{n_e \times m}, X \in \mathbb{R}^m, b \in \mathbb{R}^{n_e} \):
- #Unknowns: \( m = \sum_{j=1}^{N_m} N_{c_j}, N_{c_j} \in \{1,2,3\} \)
- #Equations: \( n_e \)

Reduce “n-tuples” list (if required)

Setting \( \tilde{x} \) a solution of P2’, simulate (call propagator):
\( E_{ch}(t_f) = T(\tilde{x}, E_{ch}(t_0)) \)

Compute deviations:
\( \delta = E_{ta}^f + V_{aim} - \tilde{E}_{ch}(t_f) \)

\( \delta < \delta_{\text{convergence?}}? \)

NO \rightarrow b = b + \delta

YES \rightarrow END

\( \{P2'\} \) requires solving a linear problem \( AX = b \) iteratively, with \( A \in \mathbb{R}^{n_e \times m}, X \in \mathbb{R}^m, b \in \mathbb{R}^{n_e} \):

Linear System type | # Solutions | Resolution method
--- | --- | ---
Undetermined \((n_e > m)\) & 0 & Least Squares
Determined \((n_e = m)\) & 1 & System Inversion
Overdetermined \((n_e < m)\) & \( \infty \) & Pseudo-Inverse & Minimization of sum of norms

Problem types | Parameters | # Equations \((n_e)\)
--- | --- | ---
Transfer 2D | \( a, e_x, e_y \) | 3
Rendezvous 2D | \( a, e_x, e_y, \tau \) | 4
Transfer 3D | \( a, e_x, e_y, i, \Omega \) | 5
Rendezvous 3D | \( a, e_x, e_y, \tau, i, \Omega \) | 6
Example: ATV3-Johannes Kepler

Problem type: Rendezvous 3D
# Unknowns: 6 (TP and TV1)

Classical Approach (TN-TN-T-T strategy)

<table>
<thead>
<tr>
<th>Method</th>
<th>System inversion</th>
</tr>
</thead>
<tbody>
<tr>
<td>“n-tuples”</td>
<td>194481</td>
</tr>
<tr>
<td>#JPSIMU calls</td>
<td>8</td>
</tr>
<tr>
<td>Total DV (m/s)</td>
<td>55.27</td>
</tr>
<tr>
<td>CPU time including ephemeris generation</td>
<td>1 min 54s</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Method</th>
<th>Pseudo-Inverse</th>
<th>Min sum norms</th>
</tr>
</thead>
<tbody>
<tr>
<td>Min criterion</td>
<td>$\sum_{j=1}^{N_m}</td>
<td>\Delta V_j</td>
</tr>
<tr>
<td>“n-tuples”</td>
<td>194481</td>
<td>28561</td>
</tr>
<tr>
<td>#JPSIMU calls</td>
<td>7</td>
<td>7</td>
</tr>
<tr>
<td>Total DV (m/s)</td>
<td>55.04</td>
<td>55.03</td>
</tr>
<tr>
<td>CPU time</td>
<td>2 min</td>
<td>3 min 11 s</td>
</tr>
</tbody>
</table>
Objective: Perform Monte-Carlo simulations → Testing maneuver strategy robustness

Time consuming: Many calls to JPSIMU, JDRAGON are required!

Real world: real parameters (forces models, engines performance, vehicle features, ...). They are generated by dispersion (gaussian, uniform)

Predicted world: lack of knowledge of the real world (chaser/target restituted orbits, engines calibrations, atmospheric forecasts, ...)

JOSCAR: END-END SIMULATIONS
**JOSCAR: END-END SIMULATIONS**

- **Genoscar →** Defining the variables to disperse and the End-to-End scenario to be simulated:
  - Chaser orbit restitution
  - Target localization
  - Thrusters calibration
  - Maneuver computation
  - Rendezvous point change
  - Sequence of maneuvers
  - Atmospheric forecast

File (".xml") containing **random variables**

- **Exoscar →** Defining data required for the simulation propagation:
  - Chaser/Target vehicle
  - Force Models
  - Attitude Sequence
  - Output files of interest

**Orbit dispersions**

**Force Models**
**CONCLUSIONS**

- **Initial difficulties:**
  - **Validation:** JPSIMU has been strongly validated against PATRIUS. JDRAGON validated against DRAGON for different phasing problems (ATV, GALILEO, Mango-Picard,...) obtaining same delta-V budgets ($\pm 1$mm/s).
  - **Performance:** Expected penalty x2 (wrt Fortran) has been retrieved. For typical RDV scenarios (ATV, GALILEO), computations last from 1 to 2 min → **Acceptable.**

- **JAVA porting advantages:**
  - **Faster development:** Easier to write, compile and debug than other programming languages.
  - **Object-oriented:** Modular programs and reusable codes.
  - **Platform-independent:** Always same results (Windows, Linux, ...).
  - **Robustness:** Early checking for possible errors. Many problems are detected in advance!
  - **GUI:** Its development has been easier thanks to GENIUS and its simplified approach.
  - **New features have been proposed:** After 20 years usage feedback, obsolete functionalities have been erased, new improvements have been implemented!
Thank you for your attention! Questions?
REFERENCES


Initial orbit definition

- Timescales
  - UTC, TAI, TT, GPS, GST

- Frames
  - GCRF, CIRF, ...

- Orbital parameters
  - Keplerian
  - Cartesian
  - Circular
  - Equinoctial
  - Equatorial
  - Apogee/Perigee
  - Reentry parameters

*Relative to target also possible
Rendezvous Parameters

- Rendezvous definition
  - By date
  - By phasing duration
  - Target Orbit Nb./AOL
  - Target/Chaser Orbit Nb./AOL
- Parameters to reach
- Convergence threshold
- Aimed vector
  - Orbital parameters
  - Coelliptic orbit
  - Relative/Target (LVLH)
- Reference orbit (linearization)
**Maneuvers scenario**

- **Engines**
  - Name
  - ISP
  - Thrust

- **Fuel tanks**
  - Name
  - Thrust

- **Maneuvers to optimize**
  - Location research domain
  - Components to optimize
  - Constraints in DV
  - Constraints in angular distance

- **Trim maneuvers** (fixed in value and location)
Output

- Output files
  - Synthesis file
  - Relative ephemeris
  - Target ephemeris
  - Chaser ephemeris

- Plot XY
  - Duration
  - Chaser orbital parameters (target LVLH, QSW)

- Intermediate results
  - Progress bar
  - Remaining “n-tuples”
  - Mission plans
  - Orbital parameters deviations