Advances in Flight Dynamics Modeling and Flight Control Design by Using the DLR Flight Visualization and Flight Instruments Libraries

Daniel Milz[1] Christian Weiser[1] Franciscus L.J. van der Linden[1] Matthias Hellerer[1] Andreas Seefried[1] Tobias Bellmann[1]

[1]Institute of System Dynamics and Control, German Aerospace Center (DLR), 82234 Weßling, Germany,
{daniel.milz, christian.weiser, matthias.hellerer, andreas.seefried, tobias.bellmann}@dlr.de

Abstract

This paper presents the Flight Instruments and Flight Visualization Libraries developed within the DLR Institute of System Dynamics and Control. For the design of dynamic models and control systems, a visual evaluation of the dynamic simulation is indispensable for a successful design and test process. Especially when it comes to aircraft models, an overview of the overall dynamics is needed. Therefore, the presented libraries provide fast assembly of fully configurable and generic flight visualization tools in which the view positions as well as the setup of Primary Flight Displays can be chosen freely. This provides the visual components of a rapid prototyping environment which can be used in the development of flight dynamic models and flight control laws. Moreover, the camera views and displays can easily be reconfigured for each purpose and research focus area. Furthermore, the libraries can be used for desktop simulation, motion simulator experiments as well as flight testing on a real aircraft.
Keywords: Visualization, Flight Simulation, Flight Control

1 Introduction

Modelica has become an important language in the field of flight dynamics modeling and flight control design (Looye 2008). Moreover, the possibility of rapid prototyping within these fields has become a key technology within the development of modern aircraft (Looye 2007a). In addition to the enhancement of this rapid prototyping process for flight control laws (Looye 2007b), a visualization framework has been developed and proven helpful in supporting the design process in terms of configuration analysis, simulation and experiments.

In the field of engineering, particularly in the area of complex model and control design, visualization of physical data is indispensable for an efficient and successful proceeding. R.B. Haber already stated the importance of visualization for engineering (Haber 1990).

Until now, there is no integrated solution available for flight dynamics visualization in Modelica. Therefore, based on the Modelica Visualization Library (Bellmann 2009; Hellerer et al. 2014), the Flight Visualization and Flight Instruments Libraries have been developed. These libraries open up the possibility of a completely accessible, in-house developed aircraft visualization and simulation environment with the following main applications to be used within a rapid prototyping process:

- Desktop Simulation. The visualization models and displays can be fitted for analysis and demonstration video purposes to display relevant information directly on the screen.
- Motion Simulator Experiments.
- Flight Testing. The created flight instrument and data display views can be reused on mobile devices inside real aircraft.

Section 2 introduces the Flight Visualization Library and its components, such as the modeling of flexible structures. Following, Section 3 gives insight into the implementation of different data displays and shows exemplary setup of a standard Primary Flight Display (PFD). In a further step, Section 4 shows synergies generated through combination of both libraries. Moreover, additional features such as on-board three-dimensional virtual reality views to be used with commercial virtual reality glasses are introduced. As a conclusion of the presented work, Section 5 contains examples for easy usability of the presented framework within the process of flight dynamic modeling and flight control law development.

2 Aircraft Visualization

The DLR Flight Visualization Library is based on the DLR Visualization Library (Bellmann 2009) adding high level structures for a rapid design of flight visualization models.

Figure 1 depicts a visualization of a dynamic aircraft simulation generated with the Flight Visualization Library. In this example, a flexible aircraft structure is displayed and the forces which cause the deflection of the wing are depicted in different colors.

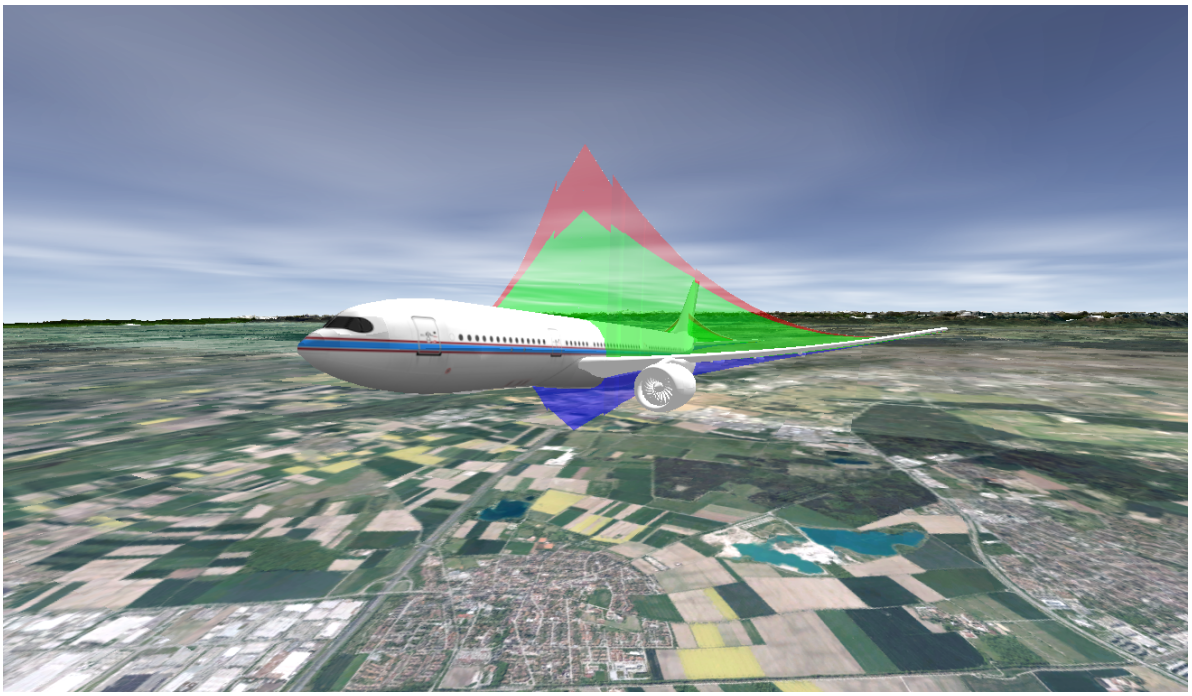

Figure 1. Example of a visualization created with the DLR Flight Visualization Library.

On the one hand, the DLR Flight Visualization Library can be used for creating stand-alone models that communicate by various internet protocols, e.g. Transmission Control Protocol (TCP) and User Datagram Protocol (UDP). Via these protocols, input data containing the current flight state and additional information on the aircraft status are received. Alternatively, a predefined trajectory can be stored internally.

On the other hand, this library can directly be integrated in various Modelica dynamic models that are e.g. models created by the DLR Flight Dynamics Library (Looye 2007a). The created models can be individualized in the following points:

- Data sources. Data in- / output using TCP / IP and UDP protocols receive data from the main flight dynamics and flight control law models.
- Terrain. Terrain files which are referenced to geodetic coordinates.
- Aircraft models. Aircraft models can be built up by one single or an assembly of multiple components.
- Camera position. Different cameras can be placed with options of switching / moving the view during the simulation.

This brings up a generic framework to visualize all known aircraft configurations as well as other vehicles that have a structure similar to aircraft such as e.g. underwater vehicles. Figure 2 shows the top level of an aircraft visualization.

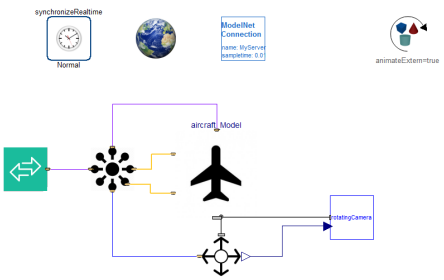

Figure 2. Top Level view of an Aircraft Visualization Model.

2.1 Aircraft Model

The aircraft model contains all visible components of the airframe structure, such as fuselage, engines, actuators. For simple simulations, one CAD file with geometry and texture is sufficient. However, the modeling of the airframe can be extended to more sophisticated setups, in which each movable component is modeled as a separate object. With this approach, control surface positions may be displayed during the simulation and can be useful as visual feedback of control action.

Flexible Models Furthermore, the results of load calculations for a flexible aircraft (Kier and Hofstee 2004) can be integrated into the aircraft visualization. For the integration of this feature, the displacement of the structural grid points obtained from the dynamic, flexible simulation is mapped to the 3D object file (Heckmann et al. 2006). One powerful method to calculate real-time deformations of free-form surfaces is the use of poly-harmonic splines defined via radial basis functions (Botsch and Kobbelt

DOI 10.3384/ecp19157481

2005).

Polyharmonic Splines A polyharmonic spline is a linear combination of radial basis functions (RBFs) denoted by:

$$f(\mathbf{x}) = \sum_{i=1}^{N} w_i \phi(|\mathbf{x} - \mathbf{c}_i|) + \mathbf{v}^T \begin{bmatrix} 1 \\ \mathbf{x} \end{bmatrix} \quad (1)$$

where $\mathbf{x} \in \mathbb{R}^3$ is the current vertex to be transformed, $\mathbf{c}_i \in \mathbb{R}^{N\times 3}$ denotes all center points, $\mathbf{w} \in \mathbb{R}^N$ represents the weights of the RBFs and $\mathbf{v} \in \mathbb{R}^{4\times 3}$ denotes the weights of the polynomial. In this case, the RBF $\phi(x)$ is defined as $\phi(x) = x^3$.

The function $f(\mathbf{x})$ calculates the displacement of a point on a flexible element. The absolute position is thus calculated as $\mathbf{x}_{\text{displaced}} = \mathbf{x} + f(\mathbf{x})$. The center points are reference points of which the deflection is known. This deflection is transformed into the weights w_i and inserted into the function.

Implementation During simulation time the calculation of the object displacements are GPU accelerated and therefore sufficiently high frame rates and smooth movement display are achieved. The GPU acceleration is implemented by adding the polyharmonic spline calculation for the vertices into the corresponding shader that iterates over all vertices. This is an essential tool for visualizing e.g. reactions to gust and turbulence and qualitative assessment of load alleviation.

2.2 Environment/ Terrain

The terrain modeling supports texture files with varying resolutions. The terrain files are mapped to geodetic longitudes and latitudes. The terrain is created using the OpenSceneGraph (OSG) Virtual Planet Builder (Pordes et al. 2007) by combining georeferenced digital elevation models (DEM) data with texture data. The Virtual Planet Builder then generates an OSG based terrain database with several levels of detail (Hellerer et al. 2014).

For simulation of landing maneuvers and collision detection, feedback from the visualization is fed back into the dynamic model in order to achieve accurate visualization results for touchdown and ground contact.

3 Cockpit Visualization

Besides a Modelica library for the external visualization of aircraft, a library for internal visualization aiming at cockpit displays is available through the DLR Flight Instruments Library. This library includes all established cockpit displays as head-up and head-down instruments and a framework for the 3-dimensional realization of virtual reality cockpits. This virtual reality framework is introduced in Section 4.1.

The two-dimensional cockpit displays are suitable for various applications including overall flight system testing, design of flight control laws, additional displays for real flight tests and many more. The main displays inside an aircraft are the Primary Flight Display (PFD), the Navigation Display (NAV) and the Electronic Centralized Aircraft Monitoring (ECAM) or Engine Indication and Crew Alerting System (EICAS). Those components consist of a set of different generic elements such as linear bars and dial gauges. In addition, aerospace related elements, e.g. an artificial horizon display, are built up.

Moreover, there is an interface to external databases and APIs included. Those may provide additional information such as navigational aids and the current or simulated weather.

3.1 General Components

The library does not only consist of high level and aircraft specific display instruments but also of low level and general use displays: A dial gauge and a linear bar instrument.

Dial Gauge The dial gauge is a standard instrument not only in terms of aerospace. Within aerospace applications it is mainly used in order to display engine data or information related to the flight control system. Figure 3 visualizes different engine parameter gauges. In addition to the marking of desired and critical domain regions for the displayed value, a change of background color gives a visual alert in case of a value exceeding its limits. The limit parameters can be adjusted quickly using a GUI based Modelica editor in order to support rapid prototyping use in experimental setups.

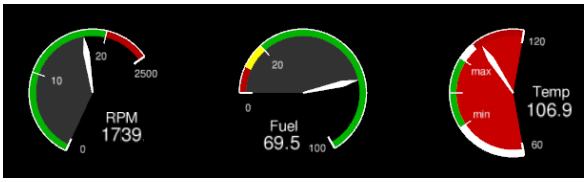

Figure 3. Screenshot of different dial gauges.

Linear Bar The linear bar represents an important instrument for modern and digital flight displays. Especially digital PFDs use bar instruments for displaying airspeed and altitude. The bar includes arrows for trend values and color-based markings for validity of a signal region.

3.2 Primary Flight Display (PFD)

The primary flight display (PFD) is the most important display in an aircraft. The main component of the PFD itself is the artificial horizon that shows the flight attitude. Modern digital PFDs show the current flight attitude, velocity and altitude of the aircraft.

Artificial Horizon The artificial horizon shows the current roll attitude Φ (Phi) and pitch attitude Θ (Theta). For civil aircraft, the center of the artificial horizon is a dot representing the aircraft's nose. In general, an artificial horizon consists of a blue area, the sky, and a brown area, the earth. This is depicted in Figure 4. The Modelica implementation of the sky surface is shown in the following code snippet.

In the beginning the model and the used variables are declared.

```
model ArtificialHorizon
    // Roll and Pitch angle
    input Real phi, theta;
    // Points defining the horizon
    Real horizon[2,2];
    // Extra points to span the two areas
    Real expt[3,2];
    // Points spanning the sky surface
    Real spt[6,2];
    // Surface, DLR Visualization Library
    Face sky(points=displaysize/2 * spt);
equation
    // ...
end ArtificialHorizon;
```

This model only takes the roll and pitch angle as an input and displays a simple artificial horizon that consists of only one surface. The design of the artificial horizon corresponds to current implementations where the visualization completely fills a quadratic display. The two horizon points are calculated as a transformation of the points `[-1,0; 1,0]` around the center `[0;0]`. Subsequently, the transformation of `horizon` and `expt` is stated.

```
if abs(horizon[1, 2]) <= 1 then
    horizon[1,1] = -1;
else
    horizon[1, 1] = minmax(interpolate({
        horizon[1, 2],-1},{horizon[2, 2],
        horizon[2, 1]},sign(horizon[1, 2]))
        );
end if;

horizon[1, 2] = -tan(phi) -theta;

if abs(horizon[2, 2]) <= 1 then
    horizon[2, 1] = 1;
else
    horizon[2, 1] = minmax(interpolate({
        horizon[1, 2],horizon[1, 1]},{
        horizon[2, 2],1}, sign(horizon[2,
        2])));
end if;

horizon[2, 2] = tan(phi) -theta;

expt[1, :] = {horizon[1,1],interpolate(
    horizon[1, :],horizon[2, :], expt[1,
    1])};
expt[2, :] = {horizon[2,1],interpolate(
    horizon[1, :],horizon[2, :], expt[2,
    1])};
expt[3, :] = {1,extrapolate(horizon[1, :],
    horizon[2, :], expt[3, 1])};
```

The `minmax` command restricts the input value to an interval $[-1;1] \subset \mathbb{R}$. Furthermore, `interpolate` and `extrapolate` take two points and a single x-value as an input. The functions calculate the y-value corresponding to the x-value on a line spanned by the two points. The `sky` surface is spanned by six two-dimensional points. Those are finally computed by the following equations:

```
spt[1,1] = if horizon[1, 2] >= 1 then
    horizon[1, 1] else -1;
spt[1,2] = 1;
spt[2,1] = if horizon[2, 2] >= 1 then
    horizon[2, 1] else 1;
spt[2,2] = 1;
spt[3,1] = if spt[3, 2] >= 1 then
    extrapolate({horizon[1, 2],horizon[1,
    1]},{horizon[2, 2],horizon[2, 1]},1)
    elseif horizon[2, 2] >= 1 then horizon
    [2, 1] else 1;
spt[3,2] = minmax(extrapolate(horizon[1,
    :],horizon[2, :],1));
spt[4,1] = if spt[3, 2] <= -1 then
    extrapolate({horizon[1,2],horizon
    [1,1]},{horizon[2,2],horizon[2,1]},-1)
    else expt[2, 1];
spt[4,2] = if spt[3, 2] <= -1 then minmax(
    extrapolate(horizon[1,:],horizon
    [2,:],1)) else minmax(expt[2, 2]);
spt[5,1] = expt[1, 1];
spt[5,2] = minmax(expt[1, 2]);
spt[6,1] = if horizon[1, 2] >= 1 then
    horizon[1, 1] else -1;
spt[6,2] = minmax(horizon[1, 2]);
```

The earth surface can be implemented analogously to the sky surface.

In Figure 4 it is shown that the current PFD setup can easily be adapted to show additional information compared to standard PFDs used in commercial aircraft. In the scope of the different works on adaptive control (Lombaerts et al. 2016; Lombaerts et al. 2018) the PFD was extended to display various flight envelopes. This is illustrated by the colored bars and areas within the display.

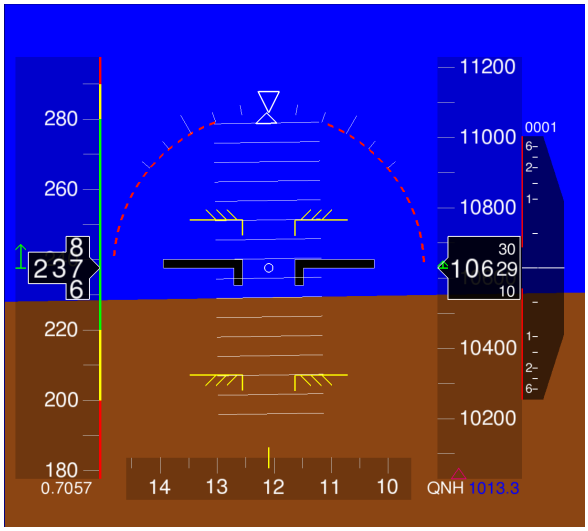

Figure 4. Primary Flight Display.

3.3 Navigation Display (NAV)

The navigation display shows a map of the surrounding navigational aids. Those include airports, VORs and many more. Furthermore, surrounding aircraft are displayed. The data can either be predefined in the model or retrieved during run-time from a database by various APIs

DOI 10.3384/ecp19157481

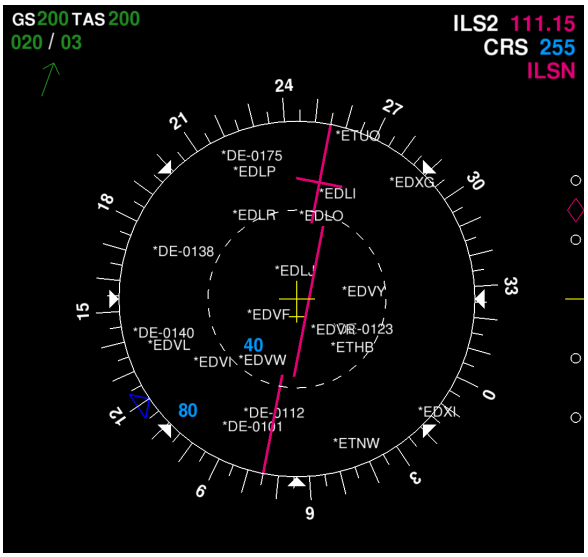

Figure 5. Navigation Display. Current simulated position near Bielefeld, Germany, while intercepting radial 255 of a radio navigation aid.

provided. Inside the DLR, a SQL database containing most of the worldwide navigational aids exists. From this database, the navigational aids within a selected radius are retrieved every second. These are stored in an array and visualized by elements provided by the DLR Visualization Library. More details on the implementation of a SQL interface is given in Section 4.3. Figure 5 illustrates an exemplary navigation display.

3.4 Additional Displays

In Figure 6, the ECAM or EICAS is an additional display that visualizes engine data such as shaft speed and fuel flow as well as the current health state of the aircraft and messages from the flight control system (FCS).

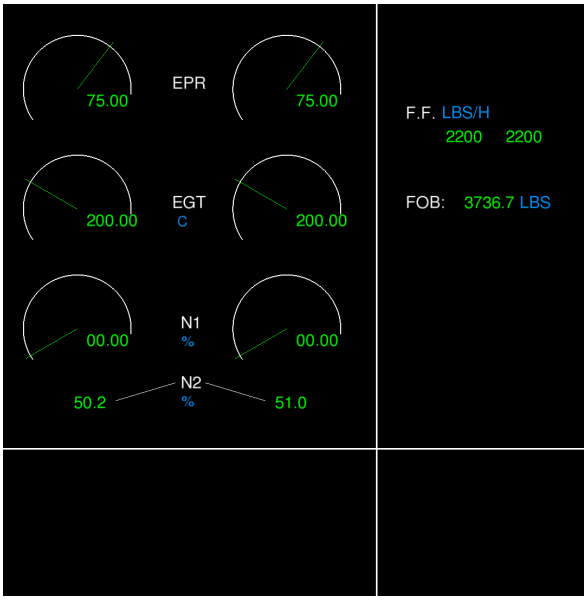

Figure 6. ECAM or EICAS display.

4 Linking of the Libraries and Additional Features

After the introduction of the two developed libraries, their linking through a common data interface is enforced in order to achieve additional benefits within the aircraft modeling and control design process. Besides, additional functions and features relevant for both libraries are introduced.

4.1 Virtual Reality Environment

A three-dimensional virtual reality cockpit perfectly fits the use for realistic pilot training and flight tests. This harmonizes with the DLR Robotic Motion Simulator described in (Bellmann et al. 2011).

For motion simulation, the visual cues are of exceptional importance. Therefore, a head mounted display (HMD) is used with a high detailed cockpit. The displays can be used within this virtual environment and be placed at the positions of the 'real' displays.

The virtual reality environment can be used in a ground based simulator as well as in combination with a motion platform, as described in Section 5.

4.2 Interaction of external and internal visualization

To add full compatibility of the two introduced libraries, a standardized interface to communicate between the models using a bus system is created. This allows the use of different predefined cockpits with individual instruments. It is not necessary to connect every display input as Modelica assumes a zero input on all unconnected bus inputs. This is in particular helpful for fast testing of new flight controllers as the user only needs to connect the needed sensor signals, leaving additional displays input blank. If required for further testing or for development of a demonstration model, the remaining sensor inputs can be connected for a fully operational cockpit.

This standardized interface allows the development of several cockpits and displays based on a single flight model. This feature has already proven to be helpful in flight tests, when the pilots shall receive a different flight display than the flight test engineers (Linden et al. 2018; Grondman et al. 2018).

To be able to use the flight instruments library with a wide range of flight models, interface converters are part of the library that convert signals from different inputs such as UDP / TCP connections directly from a real time flight computer or a Simulink model.

4.3 SQL Database Integration

In the scope of the introduced work, a generic SQL database interface is developed. M. Tiller already introduced a generic data retrieval Modelica library that is particularly designed for XML and MATLAB files (Tiller 2005). Although the use of SQL databases is promised, this solution was not available when creating this library.

Furthermore, no satisfying Modelica SQL interface is available to the point where the development of this libraries started. This motivated the creation of an own SQL interface. The relational database API is implemented in an object-oriented manner and within extenal C funcitons. Based on the `ExternalObject` parent class to manage the Modelica objects, a system of various different functions to connect, disconnect and query databases is implemented.

5 Application Examples

One major application for the previously introduced framework is the DLR Robotic Motion Simulator (RMS) (Bellmann et al. 2011). The simulator is depicted in Figure 7 and uses a robotic arm instead of a hexapod to simulate aircraft movement and accelerations. Inside the gondola, the user is wearing virtual reality glasses which provide a first person view from the pilot's seat including a visualization of the cockpit. Here, the use of the VR technology has the big advantage that no view projection is needed and the whole field of vision is covered.

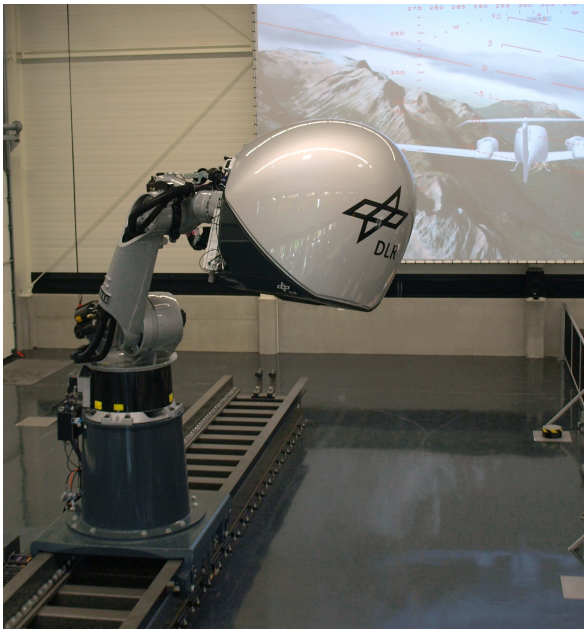

Figure 7. The DLR Robotic Motion Simulator.

A further example is the use of primary flight displays during real flight test. For this purpose, the same display as used during the design process in the desktop and simulator evaluation can be used on tablet computer and with an additional interface receive all relevant data which shall be displayed from the flight test instrumentation. This has been a major advantage during several flight test campaigns on novel flight control laws (Grondman et al. 2018) as well as current and force control for the actuation system (Linden et al. 2018). For this purpose, additional gauges displaying the actuator position and current are added to the Modelica flight displays.

Figure 8 shows the use of the Modelica Flight Instruments Library on-board the test aircraft during flight preparation. In the center of the picture a standard consumer tablet running the model can be seen. This includes a standard PFD in the top half of the picture and additional instruments and graphs for experiment relevant information in the bottom half.

Figure 8. Cockpit instruments in use for flight testing on a Cessna Citation aircraft.

6 Conclusions

This work introduced two libraries for visualization of dynamic aircraft simulations. The libraries are not only restricted to use in aircraft simulation, but can also be used for any dynamic vehicle simulation, e.g. underwater vehicles. The first part covered is the simulation of a moving vehicle in an environment, which provides the user with easy accessible visual information of the vehicle's attitude and moving direction. Secondly, instruments and displays can efficiently be used to embed important information into the visualization model or to generate a separate display. Thus, the overall process chain from model development, control design, simulation and testing is further enhanced and simplified. Additionally, all steps of the process chain are kept completely accessible for the engineer and no "black box" visualization tools are required. Both libraries are currently not distributed since development and extension are still ongoing.

References

Bellmann, Tobias (2009). "Interactive Simulations and advanced Visualization with Modelica". In: *Proceedings of the 7 International Modelica Conference Como, Italy*. Linköping University Electronic Press. DOI: 10.3384/ecp09430056.

Bellmann, Tobias, Johann Heindl, Matthias Hellerer, Richard Kuchar, Karan Sharma, and Gerd Hirzinger (2011). "The DLR Robot Motion Simulator Part I: De-

sign and setup". In: *IEEE International Conference on Robotics and Automation*. DOI: 10.1109/icra.2011.5979913.

Botsch, Mario and Leif Kobbelt (2005). "Real-Time Shape Editing using Radial Basis Functions". In: *Computer Graphics Forum* 24.3, pp. 611–621. DOI: 10.1111/j.1467-8659.2005.00886.x.

Grondman, Fabian, Gertjan Looye, Richard O. Kuchar, Q. Ping Chu, and Erik-Jan Van Kampen (2018). "Design and Flight Testing of Incremental Nonlinear Dynamic Inversion-based Control Laws for a Passenger Aircraft". In: *2018 AIAA Guidance, Navigation, and Control Conference*. American Institute of Aeronautics and Astronautics. DOI: 10.2514/6.2018-0385.

Haber, R.B. (1990). "Visualization techniques for engineering mechanics". In: *Computing Systems in Engineering* 1.1, pp. 37–50. DOI: https://doi.org/10.1016/0956-0521(90)90046-N.

Heckmann, Andreas, Martin Otter, Stefan Dietz, and José Díaz López (2006). "The DLR FlexibleBody library to model large motions of beams and of flexible bodies exported from finite element programs". In: *5th International Modelica Conference*, pp. 85–95. URL: https://elib.dlr.de/47219/.

Hellerer, Matthias, Tobias Bellmann, and Florian Schlegel (2014). "The DLR Visualization Library - Recent development and applications". In: *The 10th International Modelica Conference 2014*. Linköping Electronic Conference Proceedings. LiU Electronic Press, pp. 899–911. URL: https://elib.dlr.de/92153/.

Kier, T. and J. Hofstee (2004). "VARLOADS - Eine Simulationsumgebung zur Lastenberechnung eines voll flexiblen, freifliegenden Flugzeugs". In: *Deutscher Luft- und Raumfahrtkongress, Dresden, 20.-23. September 2004*. Vol. I & II. LIDO-Berichtsjahr=2004, monograph_id=DGLR-JT 2004-240, URL: https://elib.dlr.de/12207/.

Linden, Franciscus L. J. van der, Gertjan Looye, and Tijmen Pollack (2018). "Aircraft control using actuator current". In: *International Conference on Recent Advances in Aerospace Actuation Systems and Components 2018*, pp. 196–202. URL: https://elib.dlr.de/120314/.

Lombaerts, Thomas, Gertjan Looye, Andreas Seefried, Miguel Neves, and Tobias Bellmann (2016). "Development and Concept Demonstration of a Physics Based Adaptive Flight Envelope Protection Algorithm". In: *IFAC-PapersOnLine* 49.5. 4th IFAC Conference on Intelligent Control and Automation SciencesICONS 2016, pp. 248–253. DOI: 10.1016/j.ifacol.2016.07.121.

Lombaerts, Thomas, Gertjan Looye, Andreas Seefried, Miguel Neves, and Tobias Bellmann (2018). "Proof of concept simulator demonstration of a physics based self-preserving flight envelope protection algorithm". In: *Engineering Applications of Artificial Intelligence* 67, pp. 368–380. DOI: 10.1016/j.engappai.2017.08.014.

Looye, Gertjan H. N. (2007a). "An Integrated Approach to Aircraft Modelling and Flight Control Law Design". PhD thesis. Delft University of Technology.

Looye, Gertjan H. N. (2007b). "Rapid Prototyping Using Inversion-Based Control and Object-Oriented Modelling". In: *Lecture Notes in Control and Information Sciences*. Springer Berlin Heidelberg, pp. 147–173. DOI: 10.1007/978-3-540-73719-3_8.

Looye, Gertjan H. N. (2008). "The New DLR Flight Dynamics Library". In: *6th Modelica Conference*. URL: https://elib.dlr.de/55670/.

Pordes, Ruth et al. (2007). "The open science grid". In: *Journal of Physics: Conference Series* 78.1, p. 012057. URL: http://stacks.iop.org/1742-6596/78/i=1/a=012057.

Tiller, M (2005). "Implementation of a generic data retrieval API for Modelica". In: *4th Modelica conference*. URL: https://www.modelica.org/events/Conference2005/online_proceedings/Session7/Session7b2.pdf.

DOI 10.3384/ecp19157481