Quick and dirty I/O playing with X-Plane without "dirty" - the Tcl way

Frank Morlang
German Aerospace Center (DLR)
Lilienthalplatz 7
38108 Braunschweig, Germany
frank.morlang@dlr.de

ABSTRACT
The paper describes Tcl usage in the context of interfacing with the X-Plane flight simulator. The need for a real-time flight dynamics corrector application, taking table based aerodynamic coefficients from Computational Fluid Dynamics (CFD) model experiments to overwrite X-Plane's internal flight dynamics in the supersonic and hypersonic regime resulted in the development of an interface X-Plane package in Tcl.

1. INTRODUCTION
DLR's advanced concept for a hypersonic, suborbital, fix-winged passenger transport called SpaceLiner [1] was chosen as a use case for further examinations regarding the integration of space traffic into normal air traffic. Influences of future aircraft/spacecraft air/ground information exchange need tests and analyses under human-in-the-loop flight simulation conditions, especially if human factors aspects in emergency situations are concerned [2]. X-Plane's tool called Plane Maker was used to create a geometric shape of the SpaceLiner as a wireframe model (Figure 1).

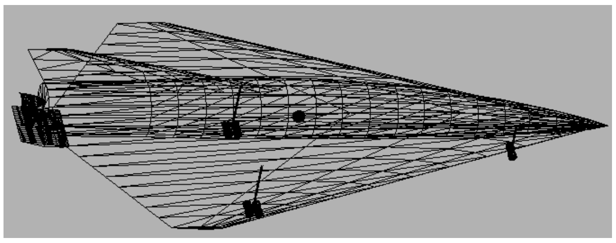


Figure 1. SpaceLiner model in X-Plane's Plane Maker tool

X-Plane derives aircraft dynamics by slicing the geometry in many small parts of the wireframe model. Taking the acting forces on each part and summing up, the actual accelerations are calculated. Integration over time reveals the velocities and double integration over time the positions ("blade element theory" process). This approach delivers high realistic accuracy in the subsonic regime (Figure 2 and Figure 3). Although X-Plane considers compressible flow effects, this realism is not reached in the supersonic / hypersonic regime (Figure 4 and Figure 5), because only an empirical Mach-divergent drag increase is modeled with an appropriate thickness ratio diamond shape representing the airfoil under supersonic conditions.

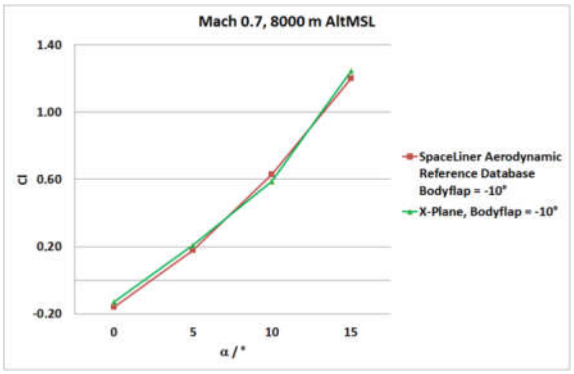


Figure 2. C_l vs. α SpaceLiner aerodynamic reference database / X-Plane model comparison

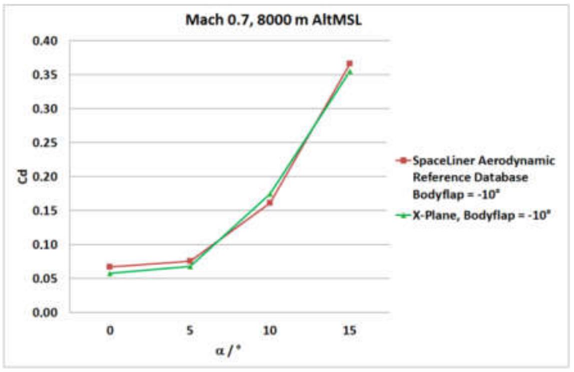


Figure 3. C_d vs. α SpaceLiner aerodynamic reference database / X-Plane model comparison

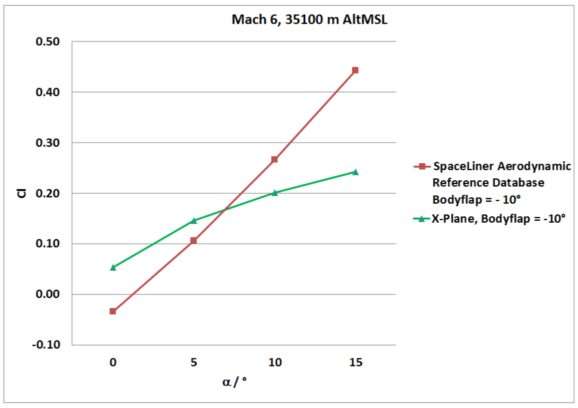


Figure 4. C_l vs. α SpaceLiner aerodynamic reference database / X-Plane model comparison

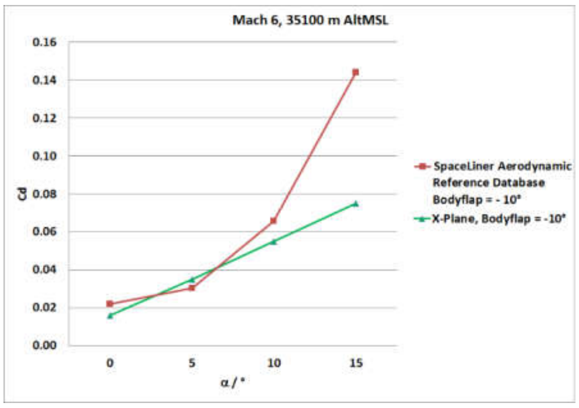


Figure 5. C_d vs. α SpaceLiner aerodynamic reference database / X-Plane model comparison

2. CHALLENGE

The challenge is to reach a constant realism throughout the whole range of regimes, subsonic, supersonic and hypersonic. In general, X-Plane's Plane Maker tool offers the modeling of additional lift and drag generating parts to be deployed in different Reynolds number ranges, thus giving the possibility to tune the flight dynamics model for a best fit of the complete reference data set. Against the background that this approach is mainly ruled by a trial-and-error character, the method can result in a very time- and cost-consuming and therefore out of scope process.

3. SOLUTION

Facing the fact that X-Plane has powerful User Datagram Protocol (UDP) input/output (i/o) capabilities, covering data fetching as well as overwriting data references with own data [3], a C_L, C_D corrector application (Figure 6) is planned. This solution will read out the speed and angle of attack (α) of the X-Plane generated flight dynamics model and overwrite its C_L and C_D values up to 100 times per second with the associated SpaceLiner aerodynamic reference database data [4] from Computational Fluid Dynamics (CFD), Calculation of Aerodynamic Coefficients (CAC)[1] & PAN AIR[2] model experiments.

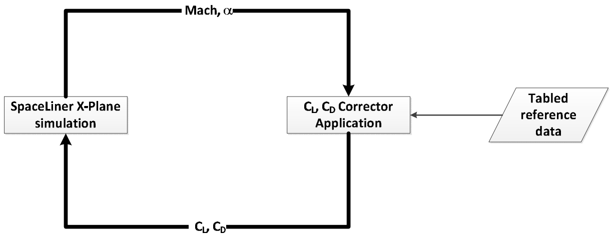

Mach, α
SpaceLiner X-Plane simulation
C_L, C_D Corrector Application
Tabled reference data
C_L, C_D

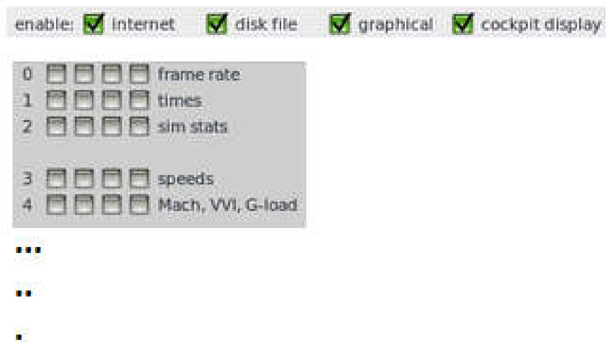
Figure 6. Corrector application

In order to create the solution's core as reusable units of code, a Tcl package called xplaneConnect was developed.

3.1 X-Plane UDP i/o

The X-Plane UDP-Message (in and out) has the following structure:

- 4 chars: "DATA"
- 1 char: internal-use INDEX-BYTE, to be "0" when sending data
- 1 int: index of the data reference (in the data output screen in X-Plane, Figure 7)
- 8 floats: up to 8 numbers of the associated data reference (-999 when not set)


enable: ✓ internet ✓ disk file ✓ graphical ✓ cockpit display

0	☐	☐	☐	☐	frame rate
1	☐	☐	☐	☐	times
2	☐	☐	☐	☐	sim stats
3	☐	☐	☐	☐	speeds
4	☐	☐	☐	☐	Mach, VVI, G-load

Figure 7. Example section of the X-Plane data output screen

3.2 The xplaneConnect package

The xplaneConnect package has the following exported functions:

- xplaneConnect_Init
- xplaneConnect_StartUdpReceiver
- xplaneConnect_DataSelect
- xplaneConnect_ProcessMessage
- xplaneConnect_SendMessage

[1] Software developed by DLR Institute of Space Systems, Dep. Space Launcher System Analysis

[2] Software developed by BOEING

The xplaneConnect_Init procedure initializes the i/o configuration by setting the used X-Plane version (10 or 11), the ip and port of the X-Plane target computer as well as the own port where messages from X-Plane should be received. The UDP socket connection is established with the help of the xplaneConnect_StartUdpReceiver procedure, which takes a callback procedure as input parameter and handles back the associated connection channel. The X-Plane data references of interest can be set with the xplaneConnect_DataSelect procedure. It needs a list of data reference numbers as input and automatically sends this data output setting to X-Plane, where the requested data references are switched to fetching via UDP mode.

3.3 The package "engine"

The xplaneConnect_ProcessMessage (Figure 8) as well as the xplaneConnect_SendMessage (Figure 9) procedures represent the "engine" of the package. It is where the full potential of Tcl's binary scan and binary format evolves. Decoding from and encoding to X-Plane's UDP message structure is realized in one line of code, thanks to the flexibility of Tcl's binary command.

Possibilities like this are often the reason for calling approaches in Tcl "quick and dirty", not realizing that it has nothing to do with "dirty".

```
proc ::xplaneConnect::xplaneConnect_ProcessMessage {argslist} {
   variable ReceivedMessage
   set inmessage [lindex $argslist 0]
   set xplaneversion [lindex $argslist 1]
   set ReceivedMessage return_from_
   append ReceivedMessage xplaneversion $xplaneversion /
   set xplanemsglength [expr {int(floor([string length $inmessage] / 36))}]
   for {set x 0} {$x < $xplanemsglength} {incr x} {
      set cutmessage [string range $inmessage [expr {5 + $x * 36}] \
            [expr {5 + $x * 36 + 35}]]
      binary scan $cutmessage iffffffff number data1 data2 data3 data4 \
            data5 data6 data7 data8
      for {set y 2} {$y < [llength $argslist]} {incr y} {
         set stopper [lindex $argslist $y]
         if {$stopper == $number} {
            append ReceivedMessage $stopper : data1 : $data1 \
                  : data2 : $data2 \
                  : data3 : $data3 \
                  : data4 : $data4 \
                  : data5 : $data5 \
                  : data6 : $data6 \
                  : data7 : $data7 \
                  : data8 : $data8 /
         }
      }
   }
   return [namespace current]::$ReceivedMessage
}
```

Figure 8. Message processing procedure

```
proc ::xplaneConnect::xplaneConnect_SendMessage {argslist} {
   variable ChannelFromXplane
   if {[llength $argslist] == 9} {
      lassign $argslist datarefnumber \
            data1 \
            data2 \
            data3 \
            data4 \
            data5 \
            data6 \
            data7 \
            data8
      set dataout "DATA0"
      append dataout [binary format iffffffff $datarefnumber \
            $data1 $data2 $data3 $data4 \
            $data5 $data6 $data7 $data8]
      puts -nonewline $ChannelFromXplane $dataout
      return 1
   } else {
      return 0
   }
}
```

Figure 9: SendMessage procedure

4. RESULTS AND DISCUSSION

The performance of the package "engine" is shown in Figure 10. It refers to a local test set-up (Table 1).

Computer	Dell Latitude E6430 8192 MB memory Intel® Core™ i7-3520M CPU @ 2.90 GHz Windows 7 Enterprise OS (SP 1)
X-Plane version	10.51, 64 bit
Tcl/Tk version	Active Tcl 8.6.9 Build 8609.2 (64 bit)

Table 1. Test set-up parameters

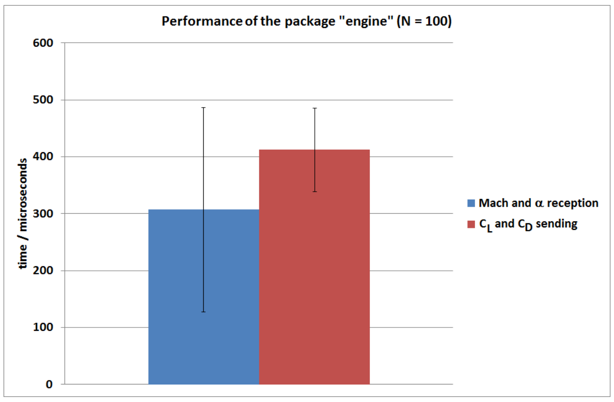


Figure 10. Performance of the package "engine"

Summarizing the xplaneConnect package "engine" performance to a rough duration estimate of about 1000 microseconds (adding Mach, α reception processing and C_L, C_D sending durations) reveals that there is enough performance "buffer" for a corrector application on a local set-up to inject C_L and C_D

corrections up to 100 times per second using table based look-up reference data.

5. OUTLOOK

The development of the C_L, C_D corrector application itself is currently in progress, using Tcllib's math::interpolate package for finding the Mach and α dependent coefficient data. The present data range for interpolation is shown in Table 2.

α	Mach								
	1.1	2.0	4.0	6.0	10.0	14.0	18.0	22.0	26.0
0.0	$x_{1,1}$	…	…	…	…	…	…	…	…
1.0	…	…	…	…	…	…	…	…	…
…	…	…	…	…	…	…	…	…	…
15.0	$x_{16,1}$	…	…	…	…	…	…	…	$x_{16,9}$

Table 2. Tabled C_L, C_D data

The current Mach and α dependencies of the SpaceLiner aerodynamic reference data set refer to a β value of 0 degrees. As soon as these data will cover a wider β range, the corrector application will be updated accordingly, already being prepared with the associated data table read in procedures.

6. REFERENCES

[1] Sippel, M., Klevanski, J., Steelant, J.: Comparative Study on Options for High-Speed Intercontinental Passenger Transports: Air-Breathing- vs. Rocket-Propelled, IAC-05-D2.4.09, October 2005

[2] Morlang, F., Ferrand, J., Seker, R.: "Why a future commercial spacecraft must be able to SWIM", 8th International Association for the Advancement of Space Safety Conference, Melbourne, Florida, 2016

[3] X-Plane 11 Desktop Manual "Data Input and Output from X-Plane",11th of June 2019. [Online]. Available: https://www.x-plane.com/manuals/desktop/#expandingx-plane.html. [Accessed 15 Aug. 2019].

[4] T. Schwanekamp, L. Morsa, G. Zuppardi and R. Molina, "SpaceLiner 7-2 Aerodynamic Reference Database", SART TN-026/2012, Bremen, 2012.