Simulierter künstlicher Horizont in Entwicklungsumgebung einer Speicherprogrammierbaren Steuerung

Stefan Gertje
Mat-Nr.: 1316492
Hochschule Fulda
36037 Fulda
Stefan.Gertje@et.hs-fulda.de

Ghaith Hamdani
Mat-Nr.: 1255935
Hochschule Fulda
36037 Fulda
Ghaith.Hamdani@et.hs-fulda.de

***Abstract*— Dieses Dokument befasst sich mit dem Design und Implementation einer künstlichen Horizonts, wie sie in Flugzeugen gefunden werden können zur Informationsgewinnung über die Raumlage der Maschine, in der Simulationsumgebung von IndraWorks von Bosch Rexroth AG.**

***Keywords*— SPS, Attitude Indicator, künstlicher Horizont**

I. Einleitung

Ein künstlicher Horizont oder auch Fluglageanzeiger zeigt Informationen über die Lage um die Querachse („Pitch“, „Nickwinkel“) und um die Längsachse („Rollwinkel“), relativ zur Erdoberfläche, in Flugzeugen und anderen Maschinen an. [1] So kann bei erschwerten Sichtbedingungen die richtige Lage der Maschine eingehalten werden. Moderne digitale Fluglageanzeiger können auch weitere Informationen zu Flughöhe, Fluggeschwindigkeit, Steigrate und Himmelsrichtung anzeigen. [2] In dem hier vorgestelltem Fluglageanzeiger werden die Lagevariablen Pitch, Roll, Height und Speed dargestellt. Ein solches Instrument in der Simulationsumgebung einer Speicherprogrammierbaren Steuerung (SPS) Entwicklungsumgebung nachzustellen kommt mit Herausforderungen und spezifischen Designentscheidungen, auf welche in diesem Dokument eingegangen wird.

II. Projektmanagement

Zur Projektübersicht wurden die in der Projektaufgabe gegebenen Anforderungen präzisiert und erweitert, woraus ein Lastenheft entstand. Aus der Anforderungsanalyse entstand ein Gantt-Diagramm zur zeitlichen Planung der Teilaufgaben. Die Entwicklung der SPS-Applikation wurde in 2 Hauptaufgaben unterteilt, eine Person sollte am Datensimulator arbeiten, welcher die notwendigen Daten zur Raumlage berechnet und über eine gemeinsam definierte Schnittstelle bereitstellt. Die andere Person arbeitet an der Visualisierung des Instruments, verarbeitet die Daten, welche er vom Datensimulator erhält, um die Visualisierung möglichst nahe dem Original zu steuern und stellt ein Bedienelement her, durch welches Benutzereingaben an die Schnittstelle gesendet werden können. Zusammenarbeit und Fortschrittsverfolgung wurde durch die Versionsverwaltungssoftware Git ermöglicht.

Lasten	Erledigt
Definition der Problemstellung:	JA
Wahl des Instruments	JA
Feststellen der notwendigen Daten	JA
Festlegen von min-max-Werten der Daten	JA
Festlegen einer Schnittstelle	JA
Instrumentenvisualisierung	JA
Feststellen notwendiger Visualisierungselemente	JA
Verhalten der Visualisierungselemente bestimmen	JA
Erstellen der Frames	JA
Statische Visualisierung erstellen	JA
Daten aus Schnittstelle abrufen	JA
Verhalten programmieren	JA
Programm mit Tests auf Fehlverhalten überprüfen	JA
Steuerelemente einfügen	JA
Visualisierung optisch ansprechend gestalten	JA
Datensimulator	JA
Festlegen der Datentypen	JA
Dynamisches Verhalten der Daten bestimmen	JA
Dynamisches Verhalten programmieren	JA
Daten zur Verfügung Stellen	JA
Plausibilitätstests entwickeln	JA
Verhalten Steuerelemente bestimmen	JA
Betriebsmodi Manuell und Automatik hinzufügen	JA

Abb. 1: Teilaufgaben zur Fortschrittsdokumentation

III. Schnittstelle

Die Schnittstelle definiert zu den Daten, welche zwischen Visualisierung und Datensimulator ausgetauscht werden die Einheiten, Wertebereiche, Namen und Datentypen. Die Daten werden nach der Richtung zwischen Datensimulator und Visualisierung gruppiert.

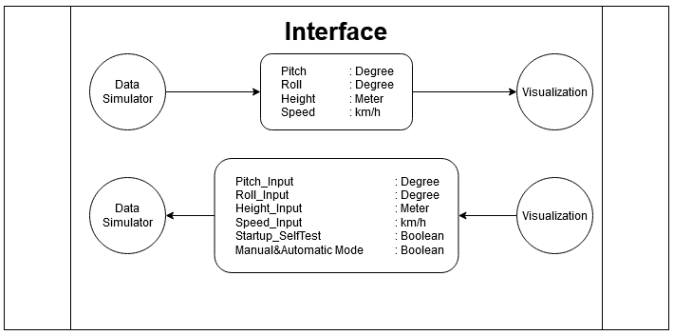

Abb. 2: Darstellung der zwischen Datensimulator und Visualisierung ausgetauschten Daten.

Der Austausch der Daten erfolgt über die globale Variablenliste „UserVarGlobal“, welche von der Entwicklungsumgebung IndraWorks bereitgestellt wird.

IV. DATENSIMULATOR

Der Datensimulator berechnet die Werte für Pitch, Roll, Flughöhe und Fluggeschwindigkeit und stellt diese der Visualisierung, über die globale Variablenliste, bereit. Der Datensimulator unterscheidet dafür zwischen 3 Zustandsmodi: Selbsttest nach dem Starten, Automatischer Modus und Manueller Modus.

A. Betriebsmodi

Selbsttest nach dem Starten: Wenn die SPS-Applikation gestartet wird und nach einem Reset werden die Variablen Pitch, Roll, Height und Speed über einen vordefinierten Wertebereich nacheinander geschrieben. Ziel ist es dem Benutzer zu ermöglichen die Funktion der Visualisierungselemente zu prüfen um mögliche Fehlersuche zu vereinfachen, ähnlich wie bei Kraftfahrzeugen die Kontrollleuchten im Cockpit bei Starten der Zündung aufleuchten um ihre Funktion zu gewährleisten.

Automatik Modus: Ähnlich wie beim Selbsttest werden die Variablen über den gleichen vordefinierten Wertebereich verändert, dies geschieht in Dauerschleife bis auf den manuellen Modus umgeschaltet wird.

Manueller Modus: Der manuelle Modus ermöglicht dem Nutzer über das Bedienpult Sollwerte für die Variablen Pitch, Roll, Height und Speed zu schreiben. Der Wert der Variablen wird über Zeit bis zum Sollwert geändert, um eine animierte Bewegung darzustellen.

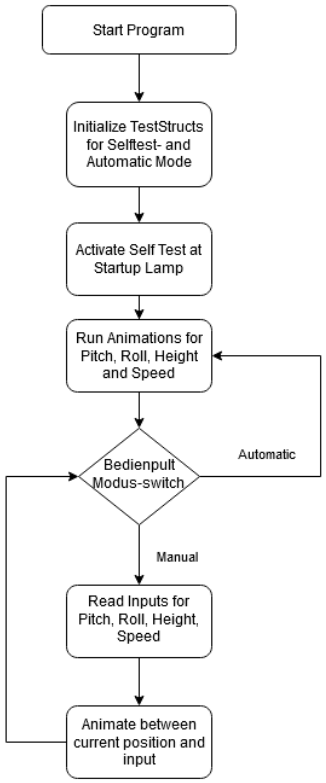

Abb. 3: Ablaufdiagramm des Datensimulators

Bei Initialisierung des Datensimulators werden die Informationen zu den durchzuführenden Tests in Variablen des Typs TestStruct geschrieben.

TestStruct		
bEnable	BOOLEAN	TRUE wenn eine Bewegung ausgeführt werden soll
startValue	INTEGER	Wert bei dem die Animation starten soll
finalValue	INTEGER	Wert bei dem die Animation enden soll
sweepInc	REAL	Wert mit dem die Animation jeden cycle incrementiert wird
testActive	BOOLEAN	TRUE wenn der zugehörige Test läuft

Abb. 4: Variablen und Datentypen der Struktur „TestStruct“

B. Testgenerator

Die Modi Selbsttest und Automatik nutzen den Funktionsbaustein TestGen, welcher in der Funktionsbausteinsprache (FUP) geschrieben wurde. Dieser verwaltet die Testzustände und Werte der Lagevariablen. Informationen zu den durchzuführenden Tests werden aus den TestStructs gelesen. Der Testgenerator verändert jede

Lagevariable einzeln mit einer Pause von 500ms dazwischen in der Reihenfolge: Pitch -> Roll -> Height -> Speed. Nachdem alle Tests durchgeführt wurden wird die Output-Variable testActive auf „FALSE“ gesetzt.

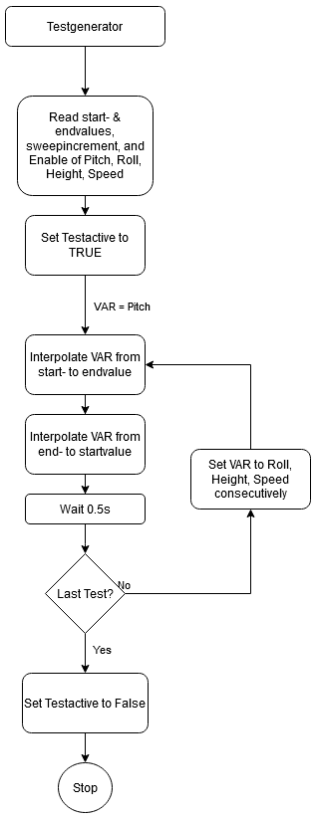

Abb. 5: Ablaufdiagramm des Funktionsbausteins TestGen

C. *Tweening*

Tweening ist eine Animationstechnik bei der eine Bewegung zwischen einem Anfangs- und Endzustand erstellt wird, indem Zwischenschritte generiert werden. [3] Dieses generieren von Zwischenschritten übernimmt in der SPS-Applikation der Funktionsbaustein Tweener, welcher in Ablaufsprache implementiert wurde. Die Eingangsvariablen dieses Funktionsbausteins sind der Anfangs- und Endwert und ein realer Wert um den inkrementiert werden soll. Ausgangsvariablen sind der momentane Wert als reale Zahl und als Ganzzahl und eine boolesche Variable „active“, welche wahr bleibt bis der Endwert erreicht wird.

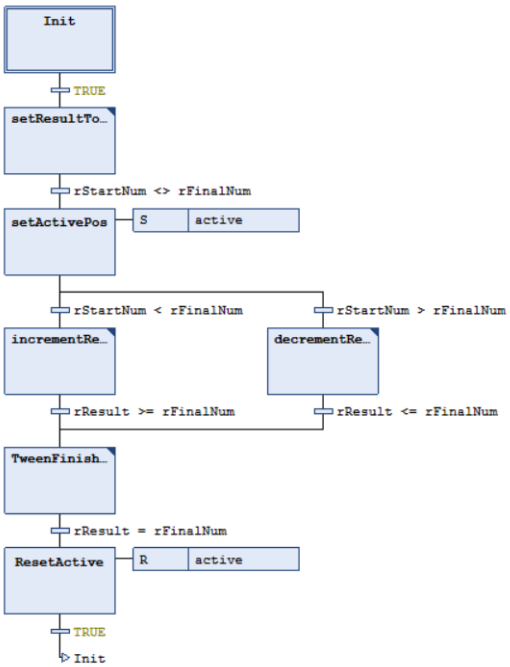

Abb. 6: Ablaufsprache des Funktionsbausteins Tweener

V. VISUALISIERUNG

A. *Instrument*

Die Visualisierung des Fluglageanzeigers soll die Werte für Pitch, Roll, Flughöhe und Fluggeschwindigkeit in einem für den Benutzer verständlichem Format und möglichst originalgetreu darstellen.

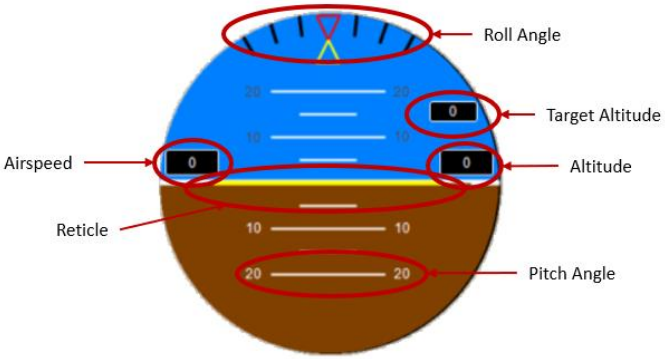

Abb. 7: Instrumentenvisualisierung mit Erklärung

Der Hintergrund des Instruments ist geteilt in eine blaue und eine braune Hälfte, welche respektiv den Himmel bzw. die Erde repräsentieren. Bei einer Änderung des Nickwinkels bewegt sich nun dieser Hintergrund nach oben oder unten, bei Änderung des Rollwinkels rotiert der Hintergrund um sein Zentrum. Konstant an ihrer Position bleiben die numerischen Anzeigen für Zielhöhe, Flughöhe und Geschwindigkeit, das gelbe Fadenkreuz („Reticle“) und das rote Dreieck sowie die schwarzen Markierungen, welche ein Maß für den Rollwinkel geben. Aufgebaut ist dieser Hintergrund aus zwei Rechtecken welche sich bewegen und drehen, und hinter einem Overlay mit einem Kreisförmigen Ausschnitt verschwinden.

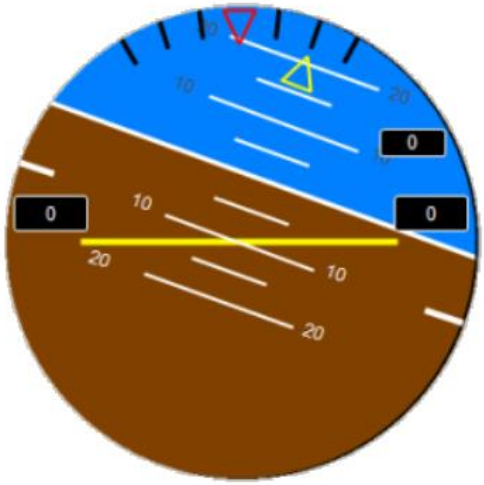

Abb. 8: Instrumentenvisualisierung bei Sinkflug mit einem Nickwinkel von -10° und einem Rollwinkel von 15°

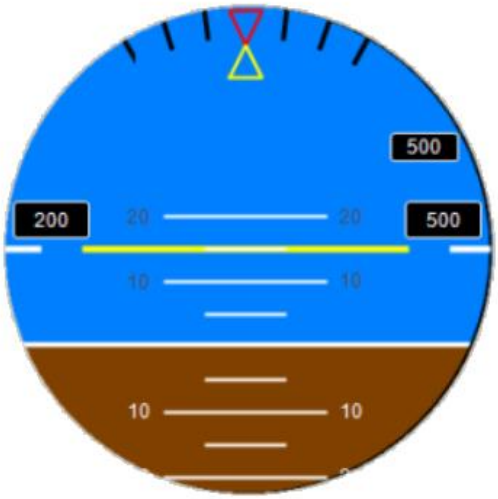

Abb. 9: Instrumentenvisualisierung bei Steigflug mit einem Nickwinkel on 15° und 0° Rollwinkel, Geschwindigkeit 200 km/h und Flughöhe 500m

A. Bedienpult

Das Bedienpult ermöglicht es dem Benutzer Sollwerte für die Lagevariablen vorzugeben, solange diese sich im vorgegeben Wertebereich befinden.

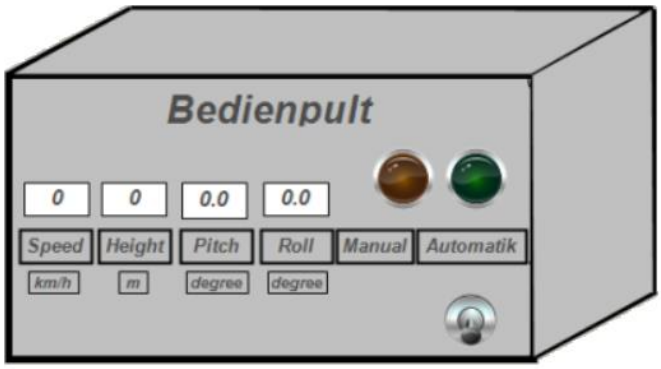

Abb. 10: Das Bedienpult der SPS-Applikation

Durch Anklicken des Zahlenwertes öffnet sich ein Pop-Up welches die Eingabe ermöglicht und den zulässigen Wertebereich beschreibt, durch das Drücken von „Eingabe“ wird dieser Wert geschrieben und an den Datensimulator weitergegeben. Während sich eine Lagevariable des Instruments ändert und während des Selbsttests zum Start, wird das Bedienpult deaktiviert, das führt dazu, dass keine neuen Werte eingegeben werden können und auch der Modus nicht geändert werden kann. Während das Instrument im Automatik Modus läuft, kann der Modus jederzeit geändert werden.

REFERENZEN

[1] Wikipedia, “Künstlicher Horizont,” 05, Mai 2004. https://de.wikipedia.org/wiki/K%C3%BCnstlicher_Horizont

[2] Garmin, “Garmin GI 275 | Electronic Flight Instrument,” *06, Februar 2024.* https://www.garmin.com/en-US/p/719027/pn/010-GI275-00

[3] S. Kench, “What is Tweening in Animation — Origins and Process Explained,” StudioBinder, 25, September 2023. https://www.studiobinder.com/blog/what-is-tweening-in-animation/