TerrainFuncs

Custom Derived Function

IADS Version 8.0 June 2012

SYMVIONICS Document SSD-IADS-156

Created June 28, 2012

Table of Contents

Table of Figures

1. Introduction

This document describes the IADS TerrainFuncs Custom Derived Function developed to assist flight test engineers with position, distance and bearing calculations.

1.1. Overview

The IADS TerrainFuncs custom derived function is provided as a Dynamic Link Library (DLL) which contains the following internal functions:

ProgId	*Description*
TerrainFuncs.EarthRadiusAtLat	Returns the radius at Geodetic Latitude
TerrainFuncs.EarthRadiusAtGeocentricLat	Returns the radius at Geocentric Latitude
TerrainFuncs.GreatCircle	Returns the shortest distance between two points
TerrainFuncs.GreatCircleAzimuth	Returns the bearing in degrees of a great circle route
TerrainFuncs.GreatCircleHav	Returns the shortest distance between two points
TerrainFuncs.GreatCircleVin	Returns the shortest distance between two points
TerrainFuncs.GreatCircleReckonLat	Returns the latitude of a destination point
TerrainFuncs.GreatCircleReckonLon	Returns the longitude of a destination point
TerrainFuncs.VincentyDirectLat	Returns the latitude of a destination point
TerrainFuncs.VincentyDirectLon	Returns the longitude of a destination point
TerrainFuncs.VincentyDirectAz2	Returns the ending azimuth of a route
TerrainFuncs.VincentyInverse	Returns the shortest distance between two points
TerrainFuncs.VincentyInverseAz1	Returns the starting azimuth of a route
TerrainFuncs.VincentyInverseAz2	Returns the ending azimuth
TerrainFuncs.DTED0_Elevation	Returns DTED0 elevation
TerrainFuncs.GTOPO30_Elevation	Returns GTOPO30 elevation

Figure 1-1 TerrainFuncs.dll Installed Functions

2. Installation

The IADS TerrainFuncs.dll is included in IADS Version 8.0 (or greater) and registers automatically upon installation of the IADS application. The following procedures are provided for other circumstances.

2.1. To register the TerrainFuncs.dll

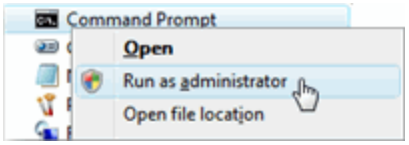
1) Copy the *TerrainFuncs.dll* file to a known directory. If you are unsure where to place it, consider the C:\Program Files\IADS directory.
2) In Windows Explorer, navigate to the directory where you copied the file and right-click on the *TerrainFuncs.dll* file and choose **Open With …**
3) Click the **Browse** button.
4) Browse to the **C:\Windows\System32** directory.
5) Select the **regsvr32.exe** file and then click the **Open** button.
6) Click **OK**. The function is now registered and available for use within IADS.

2.2. To turn off User Account Control (Windows Vista/7)

1) Open the *Control Panel* (Classic View).
2) Click on the **User Account** icon.
3) Click on **Turn User Account Control on or off**.
4) Uncheck **Use UAC to help protect your computer**.
5) Click **OK**.
6) Click on **Restart Now** to apply the changes.
7) Register the *TerrainFuncs.dll*.
8) Repeat steps 1-6 in this set of procedures to turn back on *User Account Control*.

2.3. To register a DLL as an Administrator (Windows Vista/7)

1) Right-click on the Command Prompt icon > **Run as Administrator**.



2) Enter your administrative credentials (User name, Password).
3) In the Command Prompt, navigate to the location of the dll.
4) Type: **regsvr32 TerrainFuncs.dll**
5) Press **Enter**.

3. TerrainFuncs Installed Functions

3.1. Earth Radius at Latitude

Description: Calculates the radius of the Earth at a Geodetic Latitude given the WGS84 Earth Model. Inputs are Geodetic Lat, 0.0 if Meters.

Syntax: TerrainFuncs.EarthRadiusAtLat(Latitude, zero_if_meters)

3.2. Earth Radius at Geocentric Latitude

Description: Calculates the radius of the Earth at a Geocentric Latitude given the WGS84 Earth Model. Inputs are Geocentric Lat, 0.0 if Meters.

Syntax: TerrainFuncs.EarthRadiusAtGeocentricLat(latitude, zero_if_meters)

3.3. Great Circle

Description: Calculates the shortest distance between two points on the Spherical Earth Model. Inputs are Lat1, Lon1, Lat2, Lon2, 0.0 if Meters and an optional Earth Radius from WGS84 Ellipsoid entry. If this option is not used, the default Earth Radius is the Mean as defined on Wiki: 6371008.7714.

Syntax: TerrainFuncs.GreatCircle(latitude1, longitude1, latitude2, longitude2, zero_if_meters, [Optional] Earth Radius)

3.4. Great Circle Azimuth

Description: Calculates the bearing in degrees of a Great Circle Route using a Spherical Earth Model. Inputs are Lat1, Lon1, Lat2, Lon2.

Syntax: TerrainFuncs.GreatCircleAzimuth(latitude1, longitude1, latitude2, longitude2)

3.5. Great Circle Haversine

Description: Calculates the shortest distance between two points on the Spherical Earth Model using the Haversine Formula. Inputs are Lat1, Lon1, Lat2, Lon2, 0.0 if Meters and an optional Earth Radius from WGS84 Ellipsoid entry. If this option is not used, the default Earth Radius is the Mean as defined on Wiki: 6371008.7714.

Syntax: TerrainFuncs.GreatCircleHav(latitude1, longitude1, latitude2, longitude2, zero_if_meters, [Optional] Earth Radius)

3.6. Great Circle Vincenty

Description: Calculates the shortest distance between two points on the Spherical Earth Model using the Spherical special case of the Vincenty Formula. Inputs are Lat1, Lon1, Lat2, Lon2, 0.0 if Meters and an optional Earth Radius from WGS84 Ellipsoid entry. If this option is not used, the default Earth Radius is the Mean as defined on Wiki: 6371008.7714.

Systax: TerrainFuncs.GreatCircleVin(latitude1, longitude1, latitude2, longitude2, zero_if_meters, [Optional] Earth Radius)

3.7. Great Circle Reckon Latitude

Description: Calculates the latitude of a destination point on the Spherical Earth Model. Inputs are Starting Lat, Azimuth, Range, 0.0 if Meters and an optional Earth Radius from WGS84 Ellipsoid entry. If this option is not used, the default Earth Radius is the Mean as defined on Wiki: 6371008.7714.

Systax: TerrainFuncs.GreatCircleReckonLat(Start Latitude, Azimuth, Range, zero_if_meters, [Optional] Earth Radius)

3.8. Great Circle Reckon Longitude

Description: Calculates the longitude of a destination point on the Spherical Earth Mode. Inputs are Starting Lat, Starting Lon, Azimuth, Range, 0.0 if Meters and an optional Earth Radius from WGS84 Ellipsoid entry. If this option is not used, the default Earth Radius is the Mean as defined on Wiki: 6371008.7714.

Syntax: TerrainFuncs.GreatCircleReckonLon(Start Latitude, Start Longitude, Azimuth ,Range, zero_if_meters, [Optional] Earth Radius)

3.9. Vincenty Direct Latitude

Description: Calculates the latitude of a destination point using the Vincenty Formula. Inputs are Starting Lat, Starting Azimuth, Range, 0.0 if Meters.

Syntax: TerrainFuncs.VincentyDirectLat(Start Latitude, Start Azimuth, Range, zero_if_meters)

3.10. Vincenty Direct Longitude

Description: Calculates the longitude of a destination point using the Vincenty Formula. Inputs are Starting Lat, Starting Lon, Starting Azimuth, Range, 0.0 if Meters.

Syntax: TerrainFuncs.VincentyDirectLon(Start Latitude, Start Longitude, Start Azimuth, Range, zero_if_meters)

3.11. Vincenty Direct Az2

Description: Calculates the ending azimuth in degrees of a route using the Vincenty Formula. Inputs are Starting Latitude, Starting Azimuth, Range, 0.0 if Meters.

Systax: TerrainFuncs.VincentyDirectAz2(Start Latitude, Start Azimuth, Range, zero_if_meters)

3.12. Vincenty Inverse

Description: Calculates the shortest distance between two points on the WGS84 Earth Model using the Vincenty Formula. Inputs are Lat1, Lon1, Lat2, Lon2, 0.0 if Meters.

Syntax: TerrainFuncs.VincentyInverse(Latitude1, Longitude1, Latitude2, Longitude2, zero_if_meters)

3.13. Vincenty Inverse Az1

Description: Calculate the starting azimuth of a route using the Vincenty Formula. Inputs are Lat1, Lon1, Lat2, Lon2.

Syntax: TerrainFuncs.VincentyInverseAz1(Latitude1, Longitude1, Latitude2, Longitude2)

3.14. Vincenty Inverse Az2

Description: Calculates the ending azimuth of a route using the Vincenty Formula. Inputs are Lat1, Lon1, Lat2, Lon2.

Syntax: TerrainFuncs.VincentyInverseAz2(Latitude1, Longitude1, Latitude2, Longitude2)

3.15. DTED0 Elevation

Description: Calculates DTED0 elevation using Latitude, Longitude, 0.0 if Meters, terrain file path, interp method (0 =b ilinear, any other number = nearest).

Syntax: TerrainFuncs.DTED0_Elevation(Latitude, Longitude, zero_if_meters, “terrain file path”, interp method)

3.16. GTOPO30 Elevation

Description: Calculates GTOPO30 elevation using Latitude, Longitude, 0.0 if Meters, terrain file path, interp method (0 =b ilinear, any other number = nearest).

Syntax: TerrainFuncs.GTOPOP30_Elevation(Latitude, Longitude, zero_if_meters, “terrain file path”, interp method)

4. Instructions for Use

This section will explain how to create the derived functions in IADS. Figure 4-1 displays the TerrainFuncs functions in the Parameter Defaults Table as derived parameters.

	ParameterDe...	Parameter	DataSourceT...	DataSourceArguement
150	TerrainFuncs	EarthRadiusAtLat	Derived	TerrainFuncs.EarthRadiusAtLat(Latitude, 0.0)
151	TerrainFuncs	EarthRadiusAtGeoc...	Derived	TerrainFuncs.EarthRadiusAtGeocentricLat(latitude, 0.0)
152	TerrainFuncs	GreatCircle	Derived	TerrainFuncs.GreatCircle(latitude1, longitude1, latitude2, longitude2, 0.0)
153	TerrainFuncs	GreatCircleAzimuth	Derived	TerrainFuncs.GreatCircleAzimuth(latitude1, longitude1, latitude2, longitude2)
154	TerrainFuncs	GreatCircleHav	Derived	TerrainFuncs.GreatCircleHav(latitude1, longitude1, latitude2, longitude2)

Figure 4-1 Derived Parameter Setup in the Parameter Defaults Table

To create a TerrainFuncs derived parameter:

1) In IADS, on the Dashboard click the **Configuration** button.
2) Open the **Data** folder > click **Parameter Defaults**.
3) Copy and paste an existing row of data for a parameter that is similar to the one you are creating.
4) In the *Parameter* column enter a unique parameter name.
5) In the *Data Source Type* column, select **Derived**.
6) Enter the desired *TerrainFuncs* function in the *Data Source Argument* column, for example: TerrainFuncs.GreatCircleAzimuth(latitude1, longitude1, latitude2, longitude2)
7) Click the *Save* button.