NORWEGIAN UNSIVERSITY OF SCIENCE AND TECHNOLOGY

Application Programming Interface

Njord

Espen Tolpinrud

07.06.2012
Revision: 1.0.0

This document is meant to be used as an Application Programming Interface (API) for future control system developers.

Table of Content

Introduction

It is recommended to have some basic knowledge of LabView and programming before reading this documentation.

Explanation of expressions

Cluster is a gathering of value types much like struct in other programming languages.

Matrix is a typedef 2D Array of Double
Complex Matrix is a typedef of 2D Array of Complex Numbers.

Objects

Vessel.lvclass

Labview Object.lvclass
∟ Vessel.lvclass

Implemented by Espen Tolpinrud.

The Vessel class is a representation of the craft in the control system.

Constructor summary

Name	Note
Vessel()	Creates a Vessel Object where the TCP port number is set to default, 8500.
Vessel(UInt 16 port)	Creates a Vessel Object where the TCP port number is set to be the given port
Vessel(FPGA VI Ref)	Creates a Vessel Object where the FPGA VI Ref is set. The TCP ports is set to default

Field summary

Data Type	Name	Note
1D Array of Observer.lvclass	Array Altitude Observer	Array of Observer objects for Altitude
1D Array of Controller.lvclass	Array Controllers	Array of Controller objects
1D Array of Observer.lvclass	Array Observer	Array of Observer objects for position, velocity and acceleration estimation.
Double	Buoyancy	The Buoyancy value for the vessel
1D Array of Double	CB	The Center of Buoyancy for the vessel (Maybe to be removed)
1D Array of Double	CG	The Center of Gravity for the vessel (Maybe to be removed)
Cluster of {UInt16,1D Array of Double}	Coordinate Reference	Contains the type of coordinate system to be used together with an array of offset values.
Matrix	Coriolis_RB	A Matrix with Coriolis effects for the vessel
Int32	DOF	The number of Degrees of Freedom in the system
Int32	DP Altitude Observer Index	The Index pointing to the observer object to be used from Array Altitude Observer during DP operation
Int32	DP Controller Index	The Index pointing to the controller object to be used from the Array Controllers during DP operation
Int32	DP Guidance Index	The Index pointing to the guidance object to be used from the Guidance Array during DP operation
Int32	DP Observer Index	The Index pointing to the observer object to be used from the Array Observers
1D Array of Int32	Drop Out Array	Array indicating healthy and unhealthy sensors.
Cluster of {1D	DVL	Cluster containing information from the Doppler

Array of Double, 1D Array of Int32}		Velocity Log - Value (1D Array of Double) - Signal (1D Array of Int32)
FPGA Interface	FPGA VI Reference	A Reference to the FPGA VI implemented on the NI compactRIO.
Double	Height	The height of the vessel (Maybe to be removed)
Int32	Joystick Guidance Index	The Index pointing to the guidance object to be used from the Guidance Array during Joystick Mode
Double	Length	The length of the vessel. (Maybe to be removed)
Matrix	Linear Damping	Matrix containing linear damping values for the vessel
Matrix	Mass_A	Matrix containing the Added Mass effects for the vessel
Matrix	Mass_RB	Matrix containing the Rigid Body Mass for the vessel
1D Array of Double	Max Thrust	Array with maximum thrust in each degree of freedom for the vessel.
1D Array of Double	Max_Acc	Array with maximum acceleration defined in body coordinates in each degree of freedom for the vessel.
String	Name	Name of the ROV
Matrix	NonLinear Damping	Matrix containing nonlinear damping values for the vessel.
1D Array of Double	Nu_max	Array with maximum velocity defined in body coordinates in each degree of freedom for the vessel.
1D Array of Cluster of {Int16, Int8, Int8, Int8, Int8}	Port Configuration	Array with communication information used to communicate with the NI compactRIO. The cluster contains the fields: - Baud Rate (Int16) - Data Bits (Int8) - Parity (Int8) - Stop Bits (Int8) - Flow Control (Int8)
Double	Sampling Time	The sampling time for the control loop. Determines the frequency of iteration.
Cluster of 4 {Cluster of {1D Array of Double, 1D Array of Double, 1D Array of Double, 1D Array of Double}, Cluster of { Double, Double, Double, Double, Double}, 1D Array of Double, 1D Array of Double, Cluster of {4 1D Array of	States	Contains the different states for the system - Measured (Cluster) o Eta (1D Array of Double) o Nu (1D Array of Double) o Eta_dot (1D Array of Double) o Nu_dot (1D Array of Double) - Estimated (Cluster) o Eta (1D Array of Double) o Nu (1D Array of Double) o Eta_dot (1D Array of Double) o Nu_dot (1D Array of Double) - Desired (Cluster) o Eta (1D Array of Double) o Nu (1D Array of Double) o Eta_dot (1D Array of Double)

Double}}		o Nu_dot (1D Array of Double) - Previous Desired (Cluster) o Eta (1D Array of Double) o Nu (1D Array of Double) o Eta_dot (1D Array of Double) o Nu_dot (1D Array of Double) - Tau (1D Array of Double) - Rpm (1D Array of Double) - Altitude (Cluster) o Measured o Estimated o Desired o Previous Desired o Approximated - Attitude o Theta Measured o Theta Estimated o Omega Measured o Omega Estimated
Supervisor.lvclass	Supervisor	A Supervisor object used to communicate with a user interface and ensure correct behavior in the system.
Thrust Allocation.lvclass	Thrust Allocation	A Thrust Allocation object used to keep track of the thrusters for the vessel
Int32	Tracking Altitude Observer Index	The Index pointing to the observer object for altitude to be used from the Array Observer Altitude during Tracking
Int32	Tracking Controller Index	The Index pointing to the controller object to be used from the Array Controllers during Tracking
Int32	Tracking Guidance Index	The Index pointing to the guidance object to be used from the Guidance Array during Tracking
Int32	Tracking Observer Index	The Index pointing to the observer object to be used from the Array Observers during Tracking
1D Array of Sensor.lvclass	Used Sensors	Array containing all sensors used by the vessel.
Double	Width	The width of the vessel. (Maybe to be removed)

Methods summary

Data Type	Name with input arguments	Note
Void	Add Element in Array Altitude Guidance{Guidance.lvclass new Altitude Guidance	Inserts a new altitude guidance object in the array.

Int32	Add Element in Array Altitude Observer (Observer.lvclass new Altitude Observer)	Inserts a new Altitude Observer object in the Array Altitude Observer and returns the index of the new element.
Int32	Add Element in Array Controller (Controller.lvclass new Controller)	Inserts a new Controller object in the Array Controllers and returns the index of the new element.
Int32	Add Element in Array Guidance (Guidance.lvclass new Guidance)	Inserts a new Guidance object in the Array Guidance and returns the index of the new element.
Int32	Add Element in Array Observers (Observer.lvclass new Observer)	Inserts a new Observer object in the Array Observers and returns the index of the new element.
Void	Add Element in Available Sensors (Sensor.lvclass new Sensor	Inserts a Sensor object in the Available Sensors array
Void	Add Element in Used Sensors (Sensor.lvclass new Sensor)	Inserts a Sensor object in the Used Sensors array.
Void	Control Structure ()	Performs an iteration of the control loop. Must be placed in an outer loop structure in order to work properly.
Void	Discrete to Continuous (Void)	Transforms the angels in estimate and desired states from discrete to continuous.
Int32	Find Controller by Name (String name)	Finds the index of the controller with the specified name in the controller array. If not present the method returns -1.
Int32	Find Observer by Name (String name)	Finds the index of the observer with the specified name in the observer array. If not present the method returns -1.
Int32	Find Sensor by ID (String Sensor ID)	Searches through the Used Sensors array and returns the Index of the desired Sensor, or -1 if no match is found.
Observer.lvclass	Get Active Altitude Observer ()	Returns the Altitude Observer object for the active task. Default task is DP operation.
Controller.lvclass	Get Active Controller ()	Returns the Controller object for the active task. Default task is DP operation.
Guidance.lvclass	Get Active Guidance ()	Returns the Guidance object for the active task. Default task is DP operation.
Observer.lvclass	Get Active Observer ()	Returns the Observer object for the active task. Default task is DP operation.
Void	Measurement Substitution (Void)	Substitutes the angle measured state with the ones from MTi.
Void	Read Configuration File ()	Initializes the invoker and makes the system ready for execution.

4 1D Array of Double	Read FPGA FIFO ()	Reads the FIFO stack on the NI compactRIO and return arrays with the sensor values. NB! This method is only to be used together with a NI compactRIO unit. Use “Read ROV String_SIL” for non NI compactRIO setup. Return values is: - MRU - MT_DATA - Signals out - Dvl_m
2 1D Array of Double	Read ROV String_SIL (String ROV string, String NAVI string)	Reads the sensor strings and returns the sensor values. This is a substitution for the Read FPGA FIFO () for setup with no NI compactRIO setup. Return Values is: - Signals out - Dvl_m
Void	Signal Processing (1D Array of Double MRU, 1D Array of Double MT_DATA, 1D Array of Double Signals in, 1D Array of Double dvl_m)	Translates the sensor values into position and velocity measurements. The Sensors health is checked. State variables are set in the invoking vessel object.
Void	Store Active Altitude Observer (Observer.lvclass observer)	Replaces the Altitude Observer object for the active task. Updates will then be carried on in the system.
Void	Store Active Controller (Controller.lvclass controller)	Replaces the Controller object for the active task. Updates will then be carried on in the system.
Void	Store Active Guidance (Guidance.lvclass guidance)	Replaces the Guidance object for the active task. Updates will then be carried on in the system.
Void	Store Active Observer (Observer.lvclass observer)	Replaces the Observer object for the active task. Updates will then be carried on in the system.
Void	Write to cRIO ()	Create and write the telebuf to the NI compactRIO FIFO stack. NB! To not use this unless a NI compactRIO unit is connected. Use “Write to cRIO_SIL” instead.
1D Array of UInt8	Write to cRIO_SIL ()	Create the telebuf and send it as output. This is a substitute for the “Write to cRIO”.

Supervisor.lvclass

Labview Object.lvclass

∟ Supervisor.lvclass

Implemented by Espen Tolpinrud.

The Supervisor object works as a communication administrator and execution flow manager.

Constructor Summary

Name	Note
Supervisor()	Default constructor to create a supervisor object.

Field Summary

Data Type	Name	Note
Boolean	Auto Depth	Used to give message about running auto depth in joystick control.
Boolean	Auto Depth Tele	Used to give message about running auto depth in joystick control. Message from ROV control panel.
Boolean	Auto Heading	Used to give message about running auto heading in joystick control. Message from Joystick.
Boolean	Auto Heading Tele	Used to give message about running auto heading in joystick control. Message from ROV control panel.
Cluster of { Boolean, Int8, Double, Double, Double, Double, Double}	Altitude	Cluster containing: - Altimeter Active (Boolean) - Beam Select (Int8) - Measured (Double) - Estimated (Double) - Desired (Double) - Previous Desired (Double) - Approximated (Double)
Cluster of { Boolean, Boolean, Boolean, Int32 }	Altitude Cluster	Cluster containing: - Use Integrator (Boolean) - Use FeedForward (Boolean) - Use Reference Model (Boolean) - Desired Altitude
Cluster of { 4x 1D Array of Double}	Attitude	Cluster containing: - Theta Measured (1D Array of Double) - Theta Estimated (1D Array of Double) - Omega Measured (1D Array of Double) - Omega Estimated (1D Array of Double)
Cluster of { Boolean}	Attitude Cluster	Cluster containing: - Initialize (Boolean)

Cluster of { Int32, Boolean, Boolean, Boolean, Boolean}	Camera Control	Cluster containing: - Camera Number (Int32) - Camera Up (Boolean) - Camera Down (Boolean) - Camera Left (Boolean) - Camera Right (Boolean)
Cluster of { 4x Boolean}	Collecting Unit	Contains: - In (Boolean) - Out (Boolean) - Rotate CW (Boolean) - Rotate CCW (Boolean)
UInt16	Control Mode	Selector for the different control modes the vessel can operate in during Joystick mode.
Boolean	Control System Active	Flag used to determine if the control loop is to be run.
UInt16	Coordinate System	Selector for the different coordinate reference systems the vessel can operate in.
Cluster of {1D Array of Double, 1D Array of Double, 1D Array of Double, 1D Array of Double}	Desired	Desired State values from the vessel.
1D Array of Double	Dvl data	Sensor values from the Doppler Velocity Log.
String	Error Msg	Contains an Error Message from the system, if everything is OK, it should say "OK".
Cluster of {1D Array of Double, 1D Array of Double, 1D Array of Double, 1D Array of Double}	Estimated	Estimated State values from the vessel.
Boolean	Exit Flag	Change when user interface demands it
UInt16	Frame Mode	Selector for the different frame modes the vessel can operate in during joystick mode

Cluster of { Boolean, Boolean, Int8, Boolean, Boolean, Cluster of {Double, Double, Double, Double}, Cluster of {Boolean, 1D Array of Double}}	Guidance Cluster	Cluster contains: - Set Origin (Boolean) - New WP (Boolean) - Types of Change (Int8) - Make Change (Boolean) - Optimal Heading (Boolean) - New SetPoint o North o East o Depth o Heading - Update Reference o Mode Change o Current Position
Int32	Iteration Number	The iteration number for the control loop
Cluster of {Cluster of {Double, Double, Double}, Cluster of {Double, Double, Double}}	Joystick Axis	Cluster with joystick axis inputs. Contains: - Axis Translation o X-Axis o Y-Axis o Z-Axis - Axis Rotation o X-Axis o Y-Axis o Z-Axis
Cluster of {Boolean, Boolean, Boolean, Boolean, Boolean, Boolean, Boolean, Boolean, Boolean, Boolean, Boolean, Boolean, Int32}	Joystick Commands	Cluster with joystick button inputs. Contains: - Button 1 (Boolean) - Button 2 (Boolean) - Button 3 (Boolean) - Button 4 (Boolean) - Button 5 (Boolean) - Button 6 (Boolean) - Button 7 (Boolean) - Button 8 (Boolean) - Button 9 (Boolean) - Button 10 (Boolean) - Button 11 (Boolean) - Button 12 (Boolean) - POV Direction (Int32)

Cluster of { 2 Double Matrix, 2 Boolean}	Kalman Filter Observer Tuning	Contains new tuning values for the Kalman Filter observer. Elements are: - R (Double Matrix) - Q (Double Matrix) - Update R (Boolean) - Update Q (Boolean) When the Booleans are set to true the corresponding matrices in the Non Linear PID object are updated to contain the values from the matrices in this cluster.
Cluster of {Boolean, Boolean, Boolean, Boolean, Boolean, Boolean, Int32, Int32}	Light Settings	Cluster with Light control inputs. Contains: - Light 1 - Light 2 - Light 3 - Light 4 - HMI 1 - HMI 2 - Light Intensity 1 - Light Intensity 2
Cluster of {Boolean, Boolean, Boolean, Boolean, Boolean, Boolean, Boolean, Boolean, Boolean, Boolean, Boolean }	Manipulator Control	Cluster with Manipulator control inputs. Contains: - Power Switch (Boolean) - Upper Arm Up (Boolean) - Upper Arm Down (Boolean) - Lower Arm Up (Boolean) - Lower Arm Down (Boolean) - Arm Left (Boolean) - Arm Right (Boolean) - Claw Open (Boolean) - Claw Close (Boolean) - Rotate Clockwise (Boolean) - Rotate Counterclockwise (Boolean)
Cluster of {1D Array of Double, 1D Array of Double, 1D Array of Double, 1D Array of Double}	Measured	Measured State values from vessel.
1D Array of Single	MT_DATA	Contains values from the MT sensor. Used in Attitude Observer.
Boolean	MTi Active	Flag to determine if MTi Observer shall run
Boolean	MTi Gyro	Flag to determine if MTi Gyro sensor shall be used.
Boolean	New WP List	Flag to notify about new WP List.

Cluster of { 4 Double Matrix, 4 Boolean}	Non Linear PID Tuning	Contains new tuning values for the Non Linear PID Controller. Elements are: - KP (Double Matrix) - KD (Double Matrix) - KI (Double Matrix) - KA (Double Matrix) - Update KP (Boolean) - Update KD (Boolean) - Update KI (Boolean) - Update KA (Boolean) When the Booleans are set to true the corresponding matrices in the Non Linear PID object are updated to contain the values from the matrices in this cluster.
Boolean	Only New HiPAP	Flag to determine if only new HiPAP measurements are to be sent to the observer.
UInt16	Operation Type	Selector which keeps track of which mode is active for the vessel the supervisor object belongs to.
Cluster of {Double, Double}	Origin	Represents the UTM coordinates for a desired origin. Contains: - North (Double) - East (Double)
Cluster of { 10x Boolean}	Power Instruments	Contains - HPR (Boolean) - Sonar 1 (Boolean) - Sonar 2 (Boolean) - Doppler (Boolean) - Laser (Boolean) - KRAFT (Boolean) - Pan/Tilt (Boolean) - Transponder (Boolean) - Still Cam (Boolean) - HD Cam (Boolean)
Boolean	Reset Controller	Flag to reset integrators in the controllers.
Boolean	Reset Observer	Flag to reset the values in the observers.
Boolean	Reset ROV	Flag to set the desired position to current estimated position.
Boolean	Reset SP	Flag used to notify if signal processing is to be reset.
1D Array of Double	RPM	RPM values for the vessel the supervisor object belongs to.
Cluster of {1D Array of Int32, 1D Array of Boolean}	Sensor Status	Contains: - Sensor Drop Out (1D Array of Int32) - Sensor Health (1D Array of Boolean)
2D Array of Double	Table of WP	An array with the waypoints to be used in tracking.
1D Array of Double	Tau_scaled	A percentage representation of the desired thrust in each Degree of Freedom.
1D Array of String	Thruster Names	A string array with the thruster names.

Cluster of{Cluster of {Double, Double, Double, Double, Double, Double, Double, Double, Double}, 2D Array of Double, Cluster of { Double, Double, Double, Double, Double, Double, Double, Double, Double}, UInt16, Boolean, Boolean, Boolean, Boolean}	Tracking Cluster	A Cluster with tracking performance settings. Contains: - Performance Behavior o % Velocity max lin to exp (Double) o % Velocity zero lin to exp (Double) o % Total distance at max velocity (Double) o Max Angular Velocity (Double) o Time to max linear velocity (Double) o Min distance at max velocity (Double) o Max linear velocity (Double) o Time to max angular velocity (Double) - Table of WP (2D Array of Double) - Tracking Options o Superior Bound (Double) o Inferior Bound (Double) o Duration No operation period between motions (Double) o Tolerance of initial angle (Double) o Distance to activation waiting function (Double) o Distance to deactivation waiting function (Double) o (2) Fixed Heading (Double) o (3) PoI N (Double) o (3) PoI E (Double) - Heading Mode (UInt16) - Start (Boolean) - Use Position (Boolean) - Pause (Boolean) - Abort (Boolean)
Boolean	Update Ports	A Boolean to notify the system about new port configurations.
1D Array of Cluster of {Int16, Int8, Int8, Int8, Int8}	Updated Port Configurations	Array with communication information used to communicate with the NI compactRIO. The cluster contains the fields: - Baud Rate (Int16) - Data Bits (Int8) - Parity (Int8) - Stop Bits (Int8) - Flow Control (Int8)
Boolean	Use Altitude Guidance	A Boolean to notify the system about using altitude guidance.
Boolean	Use Chair Joystick	A Boolean to notify the system about using the chair joystick instead of the one connected to the host pc. Yields only for 30k.

Method Summary

Data Type of Return	Name with input arguments	Note
TCP Network Reference	Receive TCP Message (TCP Network Reference ref)	Receive run-time commands through the TCP protocol. Usually from a user interface.
TCP Network Reference	Receive TCP Message Joystick (TCP Network Reference ref)	Receive Joystick commands through the TCP protocol and fills out Joystick Axis and Joystick Commands. Usually from a user interface.
TCP Network Reference	Send TCP Message (TCP Network Reference ref)	Sends status messages through the TCP protocol, usually to a user interface.

Method Details

Receive TCP Message (TCP Network Reference ref)

This method is used to listen for new inputs from the user interface. Each message contains a code and a message body.

Code	Message Body	Note
1:	1	Exit message to the system.
2:	Double North,Double East,Double Depth,Double Heading	Notification of new DP position. The coordinates should be given in NED coordinates with Heading in radians.
3:	Integer nWP,Double wp1index,Double wp1n,Double wp1e,Double wp1d;…;Double wpNindex,Double wpNn,Double wpNe,Double wpNd;	Notification of tracking operation. The whole waypoint list is sent starting with the number of waypoints, N, and then continuing with N sets of NED coordinates.

4:	3charString code,Integer value	Message for Manipulator control on the ROV. The code has to consist of 3 and only 3 characters! The valid codes are: - PSM (Power Switch Manipulator) - UA (Upper Arm) o U (Up) o D (Down) - LA (Lower Arm) o U (Up) o D (Down) - MA (Manipulator Arm) o R (Right) o L (Left) - CL (Claw) o C (Close) o O (Open) - AR (Arm Rotation) o L (Counterclockwise) o R (Clockwise) The integer values should be either 0 (for false) or 1 (for true). The true condition can however be set to any integer not equal to 0.
5:	2charString code,Integer value	Message for Camera settings on the ROV. The code has to consist of 2 and only 2 characters! The valid codes are: - CN (Camera Number) - CU (Camera Up) - CD (Camera Down) - CL (Camera Left) - CR (Camera Right) The integer value should be for all codes except CN be 0 (for false) or 1 (for true). For CN the integer value should be a whole number.

6:	2charString code,Integer value	Message for Light settings on the ROV. The code has to consist of 2 and only 2 characters! The character at the back must be an integer. The valid codes are: - L (Light) o 1 o 2 o 3 o 4 - H (Hydrargyrum medium-arc iodide light) o 1 o 2 - I (Light Intensity) o 1 o 2 The integer value should for all codes except for I be 0 (for false) or 1 (for true). For I the value should be a whole number between 0 and 127.
7:		Joystick Command String. (See details below)
8:	3charString,Double value	Message for changing options for Tracking
9:	1charString,Integer	Message for Pause or Abort Tracking.
10:	8x[Integer PortNumber,Integer BaudRate,Integer DataBits,Integer Parity,Integer StopBit,Integer FlowControl;]	Message for configuration of port connections between the system and the compactRIO. 8 port configurations are sent from the user interface.
11:	Integer value	Notification to the system to set new origin point. The integer value should be 0 (for false) or 1 (for true).
12:	Integer value	Message used to switch between using estimated and measured values in the controller.
13:	Integer value	Message used to switch frame mode during joystick operations. The integer value points to a selector position.
14:	Integer value	Message used to switch control mode during joystick operations. The integer value point to a selector position.
15:	Integer value	Message to notify the system to start the control loop. The integer value should be 0 (for false) or 1 (for true).
16:	Integer value	Message to notify the system to reset the observer. The integer value should be 0 (for false) or 1 (for true).
17:	Integer value	Message to notify the system to reset the controller. The integer value should be 0 (for false) or 1 (for true).
18:	Integer value	Message to notify the system to reset the ROV. The integer value should be 0 (for false) or 1 (for true). The controller is also reset by this call. The desired position is set to be the estimate.

19:	3charString,[1charString:1charString] [MatrixValuesString]	Message to notify the system to update the tuning matrices for a controller or observer. Which module to be updated, is given in the message.
20:	Integer Value	Message to notify the system to activate/deactivate the MTi. The integer value should be 0 (for false) or 1 (for true).
21:	Integer Value	Message to notify the system to activate/deactivate the MTi Gyro. The integer value should be 0 (for false) or 1 (for true).
22:	Integer Value	Message to notify the system to activate/deactivate the auto tuning of Omega used by the reference model. The integer value should be 0 (for false) or 1 (for true).
23:	Empty message body	Activates the joystick mode in force frame. Used as a fail-safe mechanism.
24:	Integer Value	Message to notify the system to activate/deactivate the use of the joystick on the chair (for 30k only). The integer value should be 0 (for false) or 1 (for true).
25:	Integer Value	Message to notify the system to activate/deactivate the HPR. The integer value should be 0 (for false) or 1 (for true).
26:	Integer Value	Message to notify the system to activate/deactivate the Sonar 1. The integer value should be 0 (for false) or 1 (for true).
27:	Integer Value	Message to notify the system to activate/deactivate the Sonar 2. The integer value should be 0 (for false) or 1 (for true).
28:	Integer Value	Message to notify the system to activate/deactivate the Doppler Velocity Log. The integer value should be 0 (for false) or 1 (for true).
29:	Integer Value	Message to notify the system to activate/deactivate the Laser. The integer value should be 0 (for false) or 1 (for true).
30:	Integer Value	Message to notify the system to activate/deactivate the KRAFT Manipulator Arm Control. The integer value should be 0 (for false) or 1 (for true).
31:	Integer Value	Message to notify the system to activate/deactivate the Pan/Tilt. The integer value should be 0 (for false) or 1 (for true).
32:	Integer Value	Message to notify the system to activate/deactivate the Transponder. The integer value should be 0 (for false) or 1 (for true).
33:	Integer Value	Message to notify the system to activate/deactivate the Still Camera function. The integer value should be 0 (for false) or 1 (for true).

34:	Integer Value	Message to notify the system to activate/deactivate the HD Camera. The integer value should be 0 (for false) or 1 (for true).
35:	2charString,Integer Value	Message to notify the system to activate/deactivate the Collecting Unit functions. The integer value should be 0 (for false) or 1 (for true). Codes are: - OU – Out - IN – In - CC – Counter Clockwise - CW - ClockWise
36:	Integer Values	Message to notify the system to activate/deactivate the “Only New HiPAP”. The integer value should be 0 (for false) or 1 (for true).
37:	Integer Value	Message to notify the system to reset the Signal Processing. The integer value should be 0 (for false) or 1 (for true).
38:	Empty Message body	Notifies the system to send a list of all available controllers and observers to the user interface.

Receive TCP Message Joystick (TCP Network Reference ref)

This method is used to receive joystick commands from the user interface. This has been given its own method due to frequent message receiving. By running this method in a separate thread concurrent with Receive TCP Message () no user inputs are ignored.

Code	Message Body	Note
7:	Integer value,Double X,Double Y, Double Z, Double ZRot,Integer Buttons,Integer POV	The integer value is a numeric representation of a Boolean value of whether or not the joystick is in use or not. X is the X-Axis position, Y is the Y-Axis position, Z is the Z-Axis position and the ZRot is the Z-Axis Rotation. The integer Buttons is a numeric representation of a Boolean array and has a value of $\sum_{i=0}^{k} 2^i$ where k is the number of buttons on the joystick. Each button combination then has its own unique value and no button pressed is set to 0. The POV integer gives the direction angle in degrees from 0 to 360 with step of 45 degrees. No direction is defined as -1.

Send TCP Message (1D Array of TCP Network Reference ref)

This method builds and sends a string of data to the user interface. There are separate TCP Network References for each type of data set to be sent.

The different data sets are:

- Position states
- Velocity states
- Thruster values and RPMs
- Altitude and dvl sensor
- MTi data (Not implemented yet)
- Other sensors can be added.

Thrust Allocation.lvclass

LabView Object.lvclass

∟ Thrust Allocation.lvclass

Implemented by Viktor Berg.

Constructor Summary

Name	Note
Thrust Allocation()	Default constructor to create an instance of the class.

Field Summary

Data Type	Name	Note
1D Array of Thruster.lvclass	Array Thrusters	Array of the thruster objects that belongs to the invoking thrust allocation object.
Matrix	Thrust Allocation Matrix	A matrix with thruster impact in each Degree of Freedom. Each row represents Degree of Freedom, and the columns are the thrusters. Also called T.
Matrix	Thrust Coefficients Matrix	A Matrix with the relationship between thrust in force and thruster RPMs. Also called K.

Method Summary

Data Type of Return	Name with input arguments	Note
Vessel.lvclass	Calculate Thrust(Vessel.lvclass vessel)	Converts the desired thruster forces and torques to thruster RPMs with the expression $u = K^{-1}T^{\dagger}\tau$
Void	Insert Thruster(Thruster.lvclass thruster, Int32 Index)	Insert a new thruster object in the Array Thruster. The thruster impact in each Degree of Freedom is calculated and added to the Thrust Allocation Matrix.
Void	Thrust Coefficients (1D Array of Double rpm)	The method calculates the Thrust Coefficients to be used by the Calculate Thrust method.
Void	Write Col in Thrust Allocation Matrix (	This is a private method for the Thrust Allocation object and is used to add a column in the Thrust Allocation Matrix.

Thruster.lvclass

LabView Object.lvclass

∟ Thruster.lvclass

Implemented by Espen Tolpinrud.

Constructor Summary

Name	Note
Thruster()	Default constructor to create an instance of the class.
Thruster (1D Array of Double input, String Type, String Name)	Constructor that fully initialize the object.

Field Summary

Data Type	Name	Note
1D Array of Double	Coefficients	Array with coefficients for the relationship between Force and RPM. Contains both for positive and negative RPM, and is made for second order relationship.
Double	Max RPM	A matrix with thruster impact in each Degree of Freedom. Each row represents Degree of Freedom, and the columns are the thrusters. Also called T.
String	Name	A Matrix with the relationship between thrust in force and thruster RPMs. Also called K.
1D Array of Double	Position Vector	The position vector represents the thruster position in body coordinates. First index is x, second y, third z, fourth rotation about z-axis, fifth rotation between x-y plane and z-axis. This is similar to spherical coordinates.
1D Array of Double	Rotation Vector	If thruster is Azimuth, this vector tells which axis to rotate about.
UInt16	Thruster Type	Selector of thruster type. Yields for fixed and Azimuth.

Method Summary

None

Sensor.lvclass

LabView Object.lvclass
∟ Sensor.lvclass

Implemented by Espen Tolpinrud.

This class may be unnecessary at the moment. Removal of the class is suggestible.

Constructor Summary

Name	Note
Sensor()	Default constructor to create an instance of the class.
Sensor (1D Array of String input)	Constructor that fully initialize the object.

Field Summary

Data Type	Name	Note
String	ID	The Sensor ID
String	Name	The Name of the sensor
String	PosX	The x position of the sensor from Center of Origin
String	PosY	The y position of the sensor from Center of Origin
String	PosZ	The z position of the sensor from the Center of Origin
String	RotX	The x rotation of the sensor
String	RotY	The y rotation of the sensor
String	RotZ	The z rotation of the sensor
String	Type	The type of the sensor.

Method Summary

Data Type of Return	Name with input arguments	Note
Boolean	Check Health (Int32 Drop Out)	Checks the health of the invoking sensor. NB! To be developed further.
Void	WriteAllData (1D Array of String)	Fully Initialize the object. Is used in Sensor(1D Array of String input)

Observer.lvclass

LabView Object.lvclass

∟ Observer.lvclass

Implemented by Espen Tolpinrud.

This is a template or abstract class for all observers and should never be used stand alone.

Constructor Summary

Name	Note
Observer()	Default constructor to create an instance of the class.
Observer(String input)	Constructor that sets the name of the Observer object.

Field Summary

Data Type	Name	Note
String	Observer Name	The name of the observer

Method Summary

Data Type of Return	Name with input arguments	Note
Vessel.lvclass	EstimateStates (Vessel.lvclass vessel)	A method that every child of Observer.lvclass must override. The method calculates the estimated states for the system.

Kalman Filter.lvclass

LabView Object.lvclass
∟ Observer.lvclass
∟ Kalman Filter.lvclass

Implemented by Viktor Berg.

Constructor Summary

Name	Note
Kalman Filter()	Default constructor to create an instance of the class.
Kalman Filter (Vessel.lvclass vessel)	Constructor that initializes PHI, GAMMA, and DELTA.

Field Summary

Data Type	Name	Note
Matrix	DELTA	Sector matrix for Kalman Filter.
Matrix	GAMMA	Sector matrix for Kalman Filter.
Matrix	PHI	Sector matrix for Kalman Filter.
Matrix	Q	Tuning matrix for Kalman Filter.
Matrix	R	Tuning matrix for Kalman Filter.

Method Summary

Data Type of Return	Name with input arguments	Note
2D array of Int32	Drop Out Converter (1D Array of Int32 Drop Out)	Transform the drop out vector to a measurement matrix. May not be needed anymore.
Vessel.lvclass	EstimateStates (Vessel.lvclass vessel)	Estimate states by using predictors and Kalman Filter equations.
3 1D Array of Double	Init Observed Values	Set up the measured values to be given the Kalman Filter. The output is: - y (1D Array of Double) - y_last (1D Array of Double) - tau (1D Array of Double) Note to developer: May need revising.
5 Matrix	Read Q and R	Returns DELTA, GAMMA, PHI, Q and R.

Methods Details

EstimateStates(Vessel.lvclass vessel)

//TODO: Insert equations used to calculate the estimated states.

Kalman Filter Altitude.lvclass

LabView Object.lvclass

∟ Observer.lvclass

∟ Kalman Filter.lvclass

∟ Kalman Filter Altitude.lvclass

Implemented by Espen Tolpinrud.

Constructor Summary

Name	Note
Kalman Filter Altitude()	Default constructor to create an instance of the class.

Method Summary

Data Type of Return	Name with input arguments	Note
Vessel.lvclass	EstimateStates (Vessel.lvclass vessel)	Estimates the altitude of the vessel.

Method Details

EstimateStates (Vessel.lvclass vessel)

// TODO: Insert equations for estimating the states.

Passive Observer.lvclass

LabView Object.lvclass
∟ Observer.lvclass
∟ Passive Observer.lvclass

Implemented by Espen Tolpinrud.

Constructor Summary

Name	Note
Passive Observer()	Default constructor to create an instance of the class.

Field Summary

Data Type	Name	Note
Matrix	Aw	System matrix for wave filter.
Matrix	Cw	Measurement matrix for wave filter.
Matrix	Delta	Wave damping matrix. Typical values are $\zeta_{ni} = 1.0$
Matrix	K1	Tuning matrix for wave filter. Typical values are $K_{1i}(\omega_{oi}) = -2(\zeta_{ni} - \lambda_i)\frac{\omega_{ci}}{\omega_{oi}}, \forall i = 1..6$ $K_{1(i+6)}(\omega_{oi}) = 2\omega_{oi}(\zeta_{ni} - \lambda_i), \forall i = 1..6$
Matrix	K2	Tuning matrix for observer. Typical values are $K_{2i} = \omega_{ci}$
Matrix	K3	Tuning matrix for bias estimator. Typical values $K_3 = 0.1K_4$
Matrix	K4	Tuning matrix for observer.
1D Array of Double	Lambda	Tuning variables to determine the notch. Typical value is $\lambda_i = 0.1$
Matrix	Omega	The diagonal elements are the frequency of the time periods for the waves. $\Omega = diag\left\{\frac{2\pi}{T_i}, \forall i = 1..6\right\}$ where typical values for $T_i = 5..20\ seconds$
1D Array of Double	Omega_cutoff	The cutoff frequency for the wave period. Must be larger than Omega.
Matrix	T	Time constants for the bias estimator.

Method Summary

Data Type of Return	Name with input arguments	Note
1D Array of Double	Bias Estimator (1D Array of Double $\tilde{y}$, Double dt)	Estimates the bias.
Vessel.lvclass	EstimateStates (Vessel.lvclass vessel)	Estimates the states for the system using the equations for the passive observer. Have both bias estimator and wave filter. Wave filter is optional.

Void	Initialize Matrices ()	Initializes the system matrices using tuning expressions.
1D Array of Double	Wave Estimator (1D Array of Double $\tilde{y}$, Double dt)	Wave filter for the passive nonlinear observer.

Method Details

Bias Estimator (1D Array of Double $\tilde{y}$, Double dt)

EstimateStates (Vessel.lvclass vessel)

Initialize Matrices ()

Wave Estimator (1D Array of Double $\tilde{y}$, Double dt)

Adaptive Passive Observer.lvclass

LabView Object.lvclass
∟ Observer.lvclass
∟ Passive Observer.lvclass
∟ Adaptive Observer.lvclass

Implemented by Viktor Berg.

Constructor Summary

Name	Note
Adaptive Observer()	Default constructor to create an instance of the class.

Field Summary

Data Type	Name	Note
Matrix	Gamma wave	
Matrix	K1h	
Matrix	K2h	
Matrix	K2l	
Matrix	K3h	
Matrix	K3l	
Matrix	K4h	
Matrix	K4l	
Matrix	Omega h	
Matrix	Omega l	
Matrix	Tf	

Method Summary

Data Type of Return	Name with input arguments	Note
Matrix	Adaptive Law (…)	
1D Array of Double	Bias Estimator	
Vessel.lvclass	EstimateStates (Vessel.lvclass vessel)	Estimates the states.
	Filter	
Void	Initialize Matrices ()	
1D Array of Double	Wave Estimator	

Guidance.lvclass

Labview Object.lvclass
ㄴ Guidance.lvclass

Implemented by Espen Tolpinrud.

This is a template or abstract class for all guidance modules in the control system.

Constructor Summary

Name	Note
Guidance()	Default constructor to create an instance of the class.

Field Summary

Data Type	Name	Note
Matrix	Delta	Tuning matrix for filter based reference model (Outgoing)
Matrix	Omega	Tuning matrix for filter based reference model (Outgoing)

Method Summary

Data Type of Return	Name with input arguments	Note
Vessel.lvclass	Calculate Desired States (Vessel.lvclass vessel)	Calculates the desired position and velocity for the vessel. This method must be implemented by all children of this class.

Altitude Guidance.lvclass

Labview Object.lvclass

∟ Guidance.lvclass

∟ Altitude Guidance.lvclass

Implemented by Espen Tolpinrud.

Guidance Law for altitude.

Constructor Summary

Name	Note
Altitude Guidance()	Default constructor to create an instance of the class.

Field Summary

Data Type	Name	Note
Double	K_ff	Tuning value for feed forward term
Double	K_i	Tuning value for integrator term
Double	K_p	Tuning value for proportional term

Method Summary

Data Type of Return	Name with input arguments	Note
-	Altitude GL	Private function for Altitude Guidance, used by Calculate Desired States
Vessel.lvclass	Calculate Desired States (Vessel.lvclass vessel)	Calculates the desired altitude the vessel.

DP Guidance.lvclass

Labview Object.lvclass

∟ Guidance.lvclass

∟ DP Guidance.lvclass

Implemented by Espen Tolpinrud.

Constructor Summary

Name	Note
DP Guidance()	Default constructor to create an instance of the class.

Field Summary

Data Type	Name	Note
Double	Criterium for Optimal Heading	Tuning value used in the optimal heading method (Not implemented) (Can be removed?)
1D Array of Double	Position Reference	
Cluster of {1D Array of Double, 1D Array of Double, 1D Array of Double, 1D Array of Double}	Previous Kinematics	
1D Array of Double	Restart Criteria	Tuning for the feedback method. Used to determine when to restart desired position updates.
1D Array of Double	Wait Criteria	Tuning for the feedback method. Used to determine when to wait for the vessel to catch up to the reference.

Method Summary

Data Type of Return	Name with input arguments	Note
Vessel.lvclass	Calculate Desired States (Vessel.lvclass vessel)	Calculates the desired position and velocity for the vessel.
3 1D Array of double, Boolean	Feedback (Vessel.lvclass vessel, Boolean reset_in, Boolean use_estimated_values)	Calculate the next desired step if the vessel is close enough to the current desired step. If not, the reference waits for the vessel to catch up.

Joystick Guidance.lvclass

Labview Object.lvclass
∟ Guidance.lvclass
∟ Joystick Guidance.lvclass

Implemented by Espen Tolpinrud.

Constructor Summary

Name	Note
Joystick Guidance()	Default constructor to create an instance of the class.

Field Summary

Data Type	Name	Note
UInt16	Control Mode	Selector of what control mode the joystick input is set for. Modes are Position, Velocity and Thrust.
UInt16	Control Mode Previous	Selector of what control mode the joystick input is set for from last time step
1D Array of Double	Eta Set	A copy of desired position. Helps to restrain depth changes due to uncontrolled pitch movement while driving.
Uint16	Frame Mode	Selector of what frame mode the joystick input is set for. Modes are Body, NED and cylindrical.
UInt16	Frame Mode Previous	Selector of what frame mode the joystick input is set for from last time step.
Cluster of {Boolean, Boolean, Boolean, Boolean, Boolean}	Inside_Dead_Zone	A cluster of Booleans. A cluster element is true if the input in the corresponding direction or rotation is zero i.e. inside the dead zone.
Cluster of {Boolean, Boolean, Boolean, Boolean, Boolean}	Inside_Dead_Zone_Previous	A cluster of Booleans. A cluster element is true if the input in the corresponding direction or rotation is zero i.e. inside the dead zone from last time step.
Int32	Loop count	The iteration number for the control system.
Double	Radius	The radius used in cylindrical frame mode.
Double	Radius Initially	The initially radius in cylindrical frame mode.
Boolean	Set Origin	A Boolean used to notify whether or not to set origin.
Boolean	Set Turn Point	A Boolean used to notify whether or not to set turn point.
Double	Step Depth	A standard step in depth direction. Usually used to make small corrections in the direction by button inputs or POV direction stick.
Double	Step Heading	A standard step in heading angle. Usually used to make small corrections on the angle by button inputs

		or POV direction stick.
Double	Step Surge	A standard step in surge direction. Usually used to make small corrections in the direction by button inputs or POV direction stick.
Double	Step Sway	A standard step in sway direction. Usually used to make small corrections in the direction by button inputs or POV direction stick.
Double	Theta	Used in cylindrical frame mode
Cluster of {19 Boolean}	VI Flow Criteria	The cluster is updated on each time step by the Compute Conditions method. The cluster contains - (1) CylinderMode AND (2) - (2) Out of DeadZone-Z-Axis Rot AND (3) - (3) Change ControlMode OR (10) - (4) Out of DeadZone XY-Plane - (5) Out of DeadZone XY-Plane OR Set Origin Point - (6) Set Origin Point - (7) Deactivating CylinderMode OR Initialize - (8) Deactivate CylinerMode - (9) Initialize - (10) Activating CylinderMode - (11) Set Turn pos OR (10) - (12) (11) AND CylinderMode - (13) (11) OR (1) - (14) (1) OR (16) - (15) Activating OR Deactivating CylinderMode - (16) (15) OR Initialize - (17) (5) OR (16) - (18) Out of DeadZone Z-Axis Rot OR (17) - (19) Out of DeadZone Z-Axis Rot

Method Summary

Data Type of Return	Name with input arguments	Note
Vessel.lvclass	Calculate Desired States (Vessel.lvclass vessel)	Calculates the desired position and velocity for the vessel. Input from user through joystick is used to generate the desired states.
Void	Compute Condition (Double X-Axis, Double Y-Axis, Double Z-Axis, Double Z-Rotation, Int32 Loop nr, Boolean First)	Compute the Boolean values in the cluster VI Flow Criteria.
1D Array of Double	Cylinder Kinematics Modification (1D Array of Double Max Velocity, Double Radius)	Modify the max velocity to cylinder coordinates.

1D Array of Double, Double, Double	Eta Set Update (1D Array of Double eta, Boolean condition, Boolean CylinderMode)	Updates the position reference used to maintain desired depth and heading.
Boolean	Initialize Check (Int32 Loop Counter)	Checks if the system needs to be initialized or re-initialized.

Tracking Guidance.lvclass

Labview Object.lvclass
∟ Guidance.lvclass
∟ Tracking Guidance.lvclass

Constructor Summary

Name	Note
Tracking Guidance()	Default constructor to create an instance of the class.
Tracking Guidance (Matrix Omega, Matrix Delta)	Constructor which also sets the reference model tuning matrices.

Field Summary

Data Type	Name	Note
Double	Acceptance Circle	Representation of the tolerance of position error in order to change way point.
Cluster	Previous Kinematics	States from last time step.
1D Array of Double	Restart Criteria	Tuning for the feedback method. Used to determine when to restart desired position updates.
1D Array of Double	Wait Criteria	Tuning for the feedback method. Used to determine when to wait for the vessel to catch up to the reference.

Method Summary

Data Type of Return	Name with input arguments	Note
Vessel.lvclass	Calculate Desired States (Vessel.lvclass vessel)	Calculates the desired position and velocity for the vessel. Uses a list of waypoints to perform simple and naive tracking. (To be substituted at a later point)
3 1D Array of Double	Feedback (Vessel.lvclass vessel, 1D Array of Double eta_ref, Boolean reset_in, Boolean Use Estimated Values)	Calculate the next desired step if the vessel is close enough to the current desired step. If not, the reference waits for the vessel to catch up.

Tracking STRM Guidance.lvclass

Labview Object.lvclass
∟ Guidance.lvclass
∟ Tracking Guidance STRM.lvclass

Implemented by Espen Tolpinrud.

Constructor Summary

Name	Note
Tracking Guidance STRM()	Default constructor to create an instance of the class.

Field Summary

Data Type	Name	Note
Boolean	Abort	Boolean flag used to notify if tracking action is to be aborted.
Cluster of {19 Double}	Guidance Engine Parameters	Running variables for guidance engine. (May be removed)
UInt16	Heading Mode	Selector of heading mode for tracking.
Boolean	Pause	Boolean flag used to notify if tracking is to be paused.
Cluster {8 Double}	Performance Behavior	Cluster of settings regarding ROV behavior in tracking. Values are: - % Velocity max lin to exp - % Velocity zero lin to exp - % Total dist at max velocity - % Max angular velocity - Time to max angular velocity - Min dist at max velocity - Max linear velocity - Time to max linear velocity
Double	psivar	Heading, used in variable heading case.
Boolean	Reset	Boolean flag used to notify if tracking action is to be reset.
Boolean	Start	Boolean flag used to notify if tracking action is to be started.
Cluster of {4 Double}	STRM	Cluster with values from the SynThetic Reference Model (STRM)
Boolean	Tracking Active	Boolean flag used to notify if tracking action is active.
Cluster {6 Boolean}	Tracking Engine Run-Time Logic	Cluster of Boolean flags used by the guidance engine. Values are: - Complete - Running - clrRMmemo - faststart

		- newcalc - aborting
Cluster {}	Tracking Options	Cluster with settings for the tracking. Values are: - (2) Fixed Heading - (3) PoI N - (3) PoI E - % Superior Bound - % Inferior Bound - Duration no operation period between motions - Tolerance of initial angle - Distance to activate waiting function - Distance to deactivate waiting function
Boolean	Use Depth	Boolean flag used to notify if tracking action is to use current depth.
Boolean	Waiting Function	Boolean flag used to notify if tracking action is to use waiting function.

Method Summary

Data Type of Return	**Name with input arguments**	**Note**
Vessel.lvclass	Calculate Desired States (Vessel.lvclass vessel)	Calculates the desired position and velocity for the vessel. This method must be implemented by all children of this class.
Boolean	Check if Done ()	Checks if complete and not running
Boolean	Reset Action ()	If reset, set waiting function and use depth to false.

Controller.lvclass

Labview Object.lvclass

∟ Controller.lvclass

Implemented by Espen Tolpinrud.

A template or abstract class for controller modules in the control system.

Constructor Summary

Name	Note
Controller()	Default constructor to create an instance of the class.
Controller (String name)	Constructor which sets the name of the controller.

Field Summary

Data Type	Name	Note
String	Controller Name	The name of the controller, not much used.

Method Summary

Data Type of Return	Name with input arguments	Note
Vessel.lvclass	Calculate Forces (Vessel.lvclass vessel)	Calculates the desired forces for the vessel. This method must be implemented on all children of this class.

Joystick Controller.lvclass

Labview Object.lvclass
∟ Controller.lvclass
∟ Joystick Controller.lvclass

Implemented by Espen Tolpinrud.

Controller used to control the vessel with direct thrust input

Constructor Summary

Name	Note
Controller()	Default constructor to create an instance of the class.

Field Summary

Data Type	Name	Note
Cluster of {3 Double}	PID Gains Heading	Tuning values for PID controller for heading
Cluster of {3 Double}	PID Gains Depth	Tuning values for PID controller for depth

Method Summary

Data Type of Return	Name with input arguments	Note
Vessel.lvclass	Calculate Forces (Vessel.lvclass vessel)	Gets joystick commands and sets up a desired thrust according to the input. Contains auto depth and auto heading as well.

PID Controller.lvclass

Labview Object.lvclass

∟ Controller.lvclass

∟ PID Controller.lvclass

Implemented by Espen Tolpinrud.

Constructor Summary

Name	Note
PID Controller()	Default constructor to create an instance of the class.

Field Summary

Data Type	Name	Note
Matrix	K_D	Tuning matrix for the derivative term in the controller.
Matrix	K_I	Tuning matrix for the integral term in the controller.
Matrix	K_P	Tuning matrix for the proportional term in the controller.

Method Summary

Data Type of Return	Name	Note
Vessel.lvclass	Calculate Forces (Vessel.lvclass vessel)	Calculates the desired forces for the vessel. This method must be implemented by all children of this class.
Void	Set Gains (3D Array of Double)	The tuning matrices are given through a 3D array and then set in their respective fields. K_P should be at the first index, followed by K_I in second. Last the K_D should be on the third index.

Linear PID Controller.lvclass

Labview Object.lvclass
        ∟ Controller.lvclass
                ∟ PID Controller.lvclass
                        ∟ Linear PID Controller.lvclass

Implemented by Espen Tolpinrud.

Constructor Summary

Name	Note
Linear PID Controller()	Default constructor to create an instance of the class.
Linear PID Controller (Matrix KP, Matrix KI, Matrix KD)	Constructor which gets and sets the tuning matrices.

Method Summary

Data Type of Return	Name with input arguments	Note
Vessel.lvclass	Calculate Forces (Vessel.lvclass vessel)	Calculates the desired forces for the vessel with the expression $\tau = -(K_P\tilde{\eta} + K_I\int\tilde{\eta}dt + K_D\dot{\eta})$

Non Linear PID Controller.lvclass

Labview Object.lvclass
∟ Controller.lvclass
∟ PID Controller.lvclass
∟ Non Linear PID Controller.lvclass

Implemented by Viktor Berg. Modified by Espen Tolpinrud to include optional Speed Controller.

Constructor Summary

Name	Note
Non Linear PID Controller ()	Default constructor to create an instance of the class.
Non Linear PID Controller (Matrix KP, Matrix KI, Matrix KD)	Constructor which gets and sets the tuning matrices.

Field Summary

Data Type	Name	Note
1D Array of Double	Int_limit	Used to check if the integral term shall integrate the error. If the error is larger than the Integral limit, zero is sent to the integrator.
Matrix	K_anti	Tuning matrix for the anti-wind up effect in the integral term.
Matrix	K_I_speed	Tuning matrix for the integral effect for the speed controller term
Boolean	Speed Controller	Flag to determine if to use speed controller term or not.

Method Summary

Data Type of Return	Name with input arguments	Note
Vessel.lvclass	Calculate Forces (Vessel.lvclass vessel)	Calculates the desired position and velocity for the vessel. Similar to the linear PID but also include a Feed Forward term to include non-linear effects.

Method Details

LQR Controller.lvclass

Labview Object.lvclass

∟ Controller.lvclass

∟ PID Controller.lvclass

∟ Non Linear PID Controller.lvclass

∟ LQR.lvclass

Implemented by Viktor Berg. Revised by Espen Tolpinrud.

Changes from the old system: Inherit from Non Linear Pid Controller.lvclass as the controller algorithm is the same.

This is a type of optimal control. The controller equation is based on the non-linear PID controller. The main difference between LQR and its parent Non Linear PID Controller is the automatic tuning of the matrices K_P and K_D.

Constructor Summary

Name	Note
LQR()	Default constructor to create an instance of the class.
LQR (Matrix K_i, Matrix Q, Matrix R)	Constructor which gets and sets the tuning matrices.

Field Summary

Data Type	Name	Note
Matrix	Q	Tuning matrix for Riccati equation
Matrix	R	Tuning matrix for Riccati equation

Method Summary

Data Type of Return	Name with input arguments	Note
Vessel.lvclass	Calculate Forces (Vessel.lvclass vessel)	Calculates the desired forces for the vessel. This method also calls on the parent calculate Forces () in order to find the forces.
Void	Set Gains LQR (Matrix K_I Matrix Q, Matrix R, Matrix K_anti, 1D Array of Double Integral Limit)	Sets the tuning matrices for the controller.

Sliding Mode.lvclass

Labview Object.lvclass
∟ Controller.lvclass
∟ Sliding Mode.lvclass

Implemented by Viktor Berg.

Constructor Summary

Name	Note
Sliding Mode ()	Default constructor to create an instance of the class.
Sliding Mode (Matrix K_s, Matrix K_pid, 1D Array of Double Lambda, 1D Array of Double Phi)	Constructor that gets and sets the tuning matrices.

Field Summary

Data Type	Name	Note
1D Array of Double	Int_limit	Used to check if the integral term shall integrate the error. If the error is larger than the Integral limit, zero is sent to the integrator.
Matrix	K_pid	Tuning matrix for the standard PID controller used in the algorithm.
Matrix	K_s	Tuning Matrix
1D Array of Double	Lambda	Tuning Matrix
1D Array of Double	Phi	Tuning Matrix

Method Summary

Data Type of Return	Name	Note
Vessel.lvclass	Calculate Desired States (Vessel.lvclass vessel)	Calculates the desired position and velocity for the vessel. This method must be implemented by all children of this class.
Void	Set Gains (Matrix K_s, Matrix K_pid, 1D Array of Double Lambda, 1D Array of Double Phi, 1D Array of Double Integral Limit)	Sets the tuning matrices from the corresponding input.

Backstepping.lvclass

Labview Object.lvclass

∟ Controller.lvclass

∟ Backstepping.lvclass

Not implemented yet!

Constructor Summary

Name	Note
Backstepping()	Default constructor to create an instance of the class.

Method Summary

Data Type of Return	Name with input arguments	Note
Vessel.lvclass	Calculate Desired States (Vessel.lvclass vessel)	Calculates the desired forces for the vessel.

Adaptive Backstepping.lvclass

Labview Object.lvclass

∟ Controller.lvclass

∟ Backstepping.lvclass

∟ Adaptive Backstepping.lvclass

Not implemented yet!

Stand-Alone Methods

mainVI.vi

This is a stand-alone VI used when running the system together with a compactRIO unit. The VI starts by reading a configuration file and setting up TCP/IP connections to the user interface. Four threads is then started to run concurrently sharing the same supervisor object. Mutual Exclusion mechanisms are implemented to avoid race conditions. The locks have time out limits in order to avoid dead locks. Starvation can occur, with wrong timing.

mainVI_SIL.vi

This is a stripped down version of the mainVI.vi as it is to be run with the simulator "in place". Three of the four threads are moved out to the simulator environment to ensure concurrent running. Only the "main loop" remains.

Reference Model

This is a polymorphic VI which contains a library of reference models used in different settings. The polymorphic VI can change between different VIs depending on the input, but can also be set to a fixed through a selector in the block diagram.

Method Summary

Data Type of Return	Name	Note
	Reference Model Altitude	Filter based method, scalar (Not part of polymorphic VI)
	Reference Model DP	Filter based method, vector
	Reference Model Joystick	Filter based method, vector
	Reference Model Tracking	Filter based method, vector

Kinematics

Rotation matrix $J(\Theta) = \begin{bmatrix} R(\Theta) & 0_{3\times3} \\ 0_{3\times3} & T(\Theta) \end{bmatrix}$

Data Type of Return	Name	Note
Matrix	RotationMatrix (1D Array of Double eta)	A polymorphic VI for Rotation matrices. A selector specifies which of the two RotationMatrix~ methods to be run.
Matrix	RotationMatrix6DOF (1D Array of Double eta)	Gives the rotation matrix in Euler angles for a 6 DOF system based on the position vector eta given in Euler angles.
Matrix	RotationMatrixQuaternions (1D Array of Double eta)	Gives the rotation matrix in quaternions for a 6DOF system based on the position vector eta given in quaternions.

Matrix	TransformationMatrix (1D Array of Double eta)	A polymorphic VI for Transformation matrices. A selector specifies which of the two TransformationMatrix~ methods to be run.
Matrix	TransformationMatrixEuler (1D Array of Double eta)	Gives the transformation matrix in Euler angles for a 6 DOF system based on the position vector eta given in Euler angels.
Matrix	TransformationMatrix Quaternions (1D Array of Double eta)	Gives the transformation matrix in quaternions for a 6 DOF system based on the position vector eta given in quaternions.
Matrix	SkewMatrix (1D Array of Double)	A polymorphic VI for creating a Skew matrix based on the input vector. A selector determines which of the 3DOF or 6DOF version to run.
Matrix	SkewMatrix3DOF (1D Array of Double vector)	The input vector must be of size 3. These elements are then used to create the Skew Matrix.
Matrix	SkewMatrix6DOF (1D Array of Double nu)	The input vector should be of size 6. The three last elements in the input vector are used to set up the skew matrix.
Matrix	J Library (1D Array of Double eta)	A polymorphic VI for creating the rotation matrix J. A Selector decides what form of the J matrix to be sent as an output
Matrix	J_euler (1D Array of Double eta)	Creates the rotation matrix J for 6DOF system using Euler angles.
Matrix	J_quart (1D Array of Double eta)	Creates the rotation matrix J for 6DOF system using quaternions.
Matrix	Jtrans_euler (1D Array of Double eta)	Creates the rotation matrix J for 6DOF system using Euler angels. Returns the transposed of J.
Matrix	Jtrans_quart (1D Array of Double eta)	Creates the rotation matrix J for 6 DOF system using quaternions. Returns the transposed of J.
Matrix	Jinv_euler (1D Array of Double eta)	Creates the rotation matrix J for 6DOF system using Euler angles. Returns the inverse of J.
Matrix	Jinv_euler (1D Array of Double eta)	Creates the rotation matrix J for 6 DOF system using quaternions. Returns the inverse of J.
Matrix	Jdot (1D Array of Double eta, 1D Array of Double nu)	Creates the time derived of rotation matrix by calculating $\dot{R}$ and $\dot{T}$. Only Euler angles are used for this.
1D Array of Double	EtaTransformation (1D Array of Double eta)	Transform a vector between Euler angles and quaternions using either Eta_Euler2QEta_Quart or Eta_Quart2Eta_Euler depending on transformation direction.
1D Array of Double	Eta_Euler2Eta_Quart (1D Array of Double eta)	Transforms the input vector given in NED + Euler angels to a vector given with NED + quaternions.
1D Array of Double	Eta_Quart2Eta_Euler (1D Array of Double eta)	Transforms the input vector given with NED + quaternions to a vector in NED + Euler angles.
1D Array of Double	euler2q (1D Array of Double eta)	Calculates the angles in quaternions from Euler angles. Method is taken from MSS toolbox written by Thor I. Fossen.

1D Array of Double	q2Euler(1D Array of Double eta)	Calculates the angles in Euler angels from quaternions. The method is taken from MSS toolbox written by Thor I. Fosse and translated to LabView code.

Other

Data Type of Return	Name	Note
-	Rad 2 PI-PI	Polymorphic VI for the two following methods.
Double	Rad 2 PI-PI Scalar (Double)	Converts the angle in radians to the interval $-\pi$ to π.
1D Array of Double	Rad 2 PI-PI Vector(1D Array of Double)	Converts angles in radians to the interval $-\pi$ to π.

Integrators

Data Type of Return	Name	Note
Double/1D Array of Double	Integrator Trapezoid	A Polymorphic VI for the trapezoid integrators in the system. Both the input and a selector can be used to determine the method to be used.
Double	Integrator Trapezoid scalar	A Scalar integrator using the trapezoidal rule.
1D Array of Double	Integrator Trapezoid vector	A vector integrator using the trapezoidal rule.

Signal Processing

Data Type of Return	Name	Note
	Alt_input	
	Alt_IO	
	Cocoord	
	Configure sub-VI	
	CRIO write	
	DVL_2_QUAD	
	Input converter	
	JS chair get	
	MRU_IO	
	MRU_SP	
	MRUinput subVI	
	MT_DATA_TRANS	
	MT_GET	
	MT MOD	
	NAVI_IO 2	
	NAVIpacinput subVI	
	NAVIpacinput subVI 30k	
	Origin SP	

	Port 3 IO	
	ROV_IO 2	
	ROV_IO 30k	
	ROVinput 30k subVI	
	ROVinput subVI	
	Search string	
	Set Length	
	Signal IO	
	Signal IO SIL	
	SP BC	
	SP_mod	
	Write sub-VI	

Others

Data Type of Return	**Name**	**Note**
Double Matrix	AutoParameter Guidance Omega (1D Array of Double eta_ref, 1D Array of Double eta_last)	Calculates desired omega values for the filter reference model.
1D Array of Double	Error Calculations (1D Array of Double eta, 1D Array of Double eta_des)	Used to calculate the difference between the current position and the desired position.
1D Array of Double	Eta2cont	Transform a discrete angle to continuous angle. Should not be placed such that it can miss any time step!
Double, Boolean	Get_set_point(Double setpoint, Boolean flag)	When flag is true, the current setpoint is stored and sent as output until flag is true again. (Switch on, switch off mechanism)
Matrix, Matrix	LQR	Used to calculate the tuning matrices for the LQR controller. (May move to class)
Matrix	Mass 2 Coriolis	Used to calculate the Coriolis matrix. Used in Non Linear PID and Non Linear Passive Observer.
String	Read Node Name (Tag Reference ref, String AttributeName)	Reads the attribute matching the field *AttributeName* in the tag *ref* points to. Used in the configuration file reader.
TCP Connection Reference	Reconnection Procedure	Polymorphic VI for the two following methods.
TCP Connection Reference	Reconnection Procedure Scalar (TCP Connection Reference ref, UInt16 Port)	Checks if connection is lost, if so tries to reconnect to client.
1D Array of TCP Connection Reference	Reconnection Procedure Array (1D Array of TCP Connection Reference ref, 1D Array of UInt16 ports)	Checks if the connection is lost, if so tries to reconnect to client. All connections must be reestablished or none are.

String	Scan Token Generator	Generates the scan-string used to read integer and double values from a string. Used in the String to Matrix methods.
1D Array of UInt16	Sperre thrust alloc Minerva	Converts the desired actions from the control system to a telebuffer/telegram which is to be sent to the vessel.
1D Array of UInt16	Sperre thrust alloc Neptune	Converts the desired actions from the control system to a telebuffer/telegram which is to be sent to the vessel.
Matrix	String to Matrix	A polymorphic VI that transform a String of numbers to a Matrix.
Matrix	String to Matrix Diag	Transforms a String of number to a diagonal matrix where the numbers in the String is the values on the diagonal in the matrix.
Matrix	String to Matrix Norm	Transforms a String of number to a matrix where each row in the matrix is separated by ";" in the String.
1D Array of Double	String to Vector	Transforms a String of numbers to a 1D Array of Double.
Boolean	ToggleSwitch	Used to convert a constant True/False Signal to a pulse True signal.
Boolean	True Once	Used to convert a constant True/False Signal to one pulse of True. True only once!

Configuration File

The configuration file is unique for each vessel to be used with the control system. This is what decides the setup for the system. It is therefore important that the configuration file is correct. The format on the file is set to xml as it is easy for both machines and humans to read. Keep in mind that the tags must be exact as well as the attributes.

Table of tags and attributes

Tag	Parent Tag(s)	Attributes	Notes
<Configuration>	-	-	Start tag for configuration
<Sensor>	<Configuration>	ID, Name, Type, PosX, PosY, PosZ, RotX, RotY, RotZ	Tag for initializing sensors
<Vessel>	<Configuration>	Name, Type, NDOF	Tag for vessel initialization.
<TechincalData>	<Vessel>	DOF, Length, Width, Height, Buoyancy, CenterGravity, CenterBuoyancy, TimeStep, TauMax	Tag used for vessel data
<PortConfiguration>	<Vessel>	-	Tag for initialization of communication ports for compactRIO unit
<Port>	<PortConfiguration>	ID, BaudRate, DataBits, Parity, StopBits, FlowControl	Tag for data used to set up a port for the compactRIO
<Matrix>	<Vessel>	-	Tag for matrix
<Mass>	<Matrix>	Type, Size, Values	Tag for Mass matrix
<AddedMass>	<Matrix>	Type, Size, Values	Tag for Added Mass matrix
<LinearDamping>	<Matrix>	Type, Size, Values	Tag for Linear Damping matrix
<NonLinearDamping>	<Matrix>	Type, Size, Values	Tag for Non Linear Damping matrix
<Observer>	<Vessel>	-	Tag for observer collection
<KalmanFilter>	<Observer>	Mode	Tag for Kalman Filter Observer
<KalmanFilterAltitude>	<Observer>	Mode	Tag for Altitude Kalman Filter Observer
<Passive>	<Observer>	Mode	Tag for Passive Non Linear Observer
<TuningKalman>	<KalmanFilter>, <KalmanFilterAltitude>	-	Tag for tuning fields for Kalman Filters
<TuningPassive>	<Passive>	-	Tag for tuning fields for Kalman Filters.
<R>	<TuningKalman>	Type, Size, Value	Tag for tuning matrix R
<Q>	<TuningKalman>	Type, Size, Value	Tag for tuning matrix Q

<Wave>	<TuningPassive>	TypeOmega, SizeOmega, ValueOmega, TypeDelta, SizeDelta, ValueDelta, TypeLambda, SizeLambda, ValueLambda	Tag for wave filter values
<K1>	<TuningPassive>	Type1, Size1, Value1, Type2, Size2, Value2	Tag for tuning matrix K1. Consists of two 6x6 matrices. Can be set to zero as they will be calculated in the control system by tuning equations.
<K2>	<TuningPassive>	Type, Size, Value	Tag for tuning matrix K2.
<K3>	<TuningPassive>	Type, Size, Value	Tag for tuning matrix K3
<K4>	<TuningPassive>	Type, Size, Value	Tag for tuning matrix K4
<T>	<TuningPassive>	Type, Size, Value	Tag for time constant matrix T
<Navigation>	<Vessel>	-	Tag for navigation collection
<DPOperation>	<Navigation>	Mode	Tag for DP Guidance
<Tracking>	<Navigation>	Mode	Tag for Tracking Guidance
<TrackingSTRM>	<Navigation>	Mode	Tag for Tracking Guidance with synthetic reference model
<Joystick>	<Navigation>	Mode	Tag for Joystick Guidance
<Criteria>	<DPOperation>, <Tracking>, <Joystick>	SizeRestart, Restart, SizeWait, Wait	Tag for restart and wait criteria values for guidance objects.
<Omega>	<DPOperation>, <Tracking>, <Joystick>	Type, Size, Value	Tag for reference model tuning matrix omega.
<Delta>	<DPOperation>, <Tracking>, <Joystick>	Type, Size, Value	Tag for reference model damping matrix
<Controller>	<Vessel>	-	Tag for controller collection
<PIDLinear>	<Controller>	Mode	Tag for Linear PID Controller
<NonLinearPID>	<Controller>	Mode	Tag for Non Linear PID Controller
<LQR>	<Controller>	Mode	Tag for LQR Controller
<SlidingMode>	<Controller>	Mode	Tag for Sliding Mode Controller
<TuningPID>	<PIDLinear>, <PIDNonLinear>, <LQR>	-	Tag for tuning fields for PID based controllers. These include linear, non-linear PID as well as LQR
<TuningSliding>	<SlidingMode>	-	Tag for tuning fields for Sliding Mode controller.
<KP>	<TuningPID>	Type, Size, Value	Tag for Proportional term tuning matrix
<KI>	<TuningPID>	Type, Size, Value	Tag for Integral term tuning matrix

<KD>	<TuningPID>	Type, Size, Value	Tag for Derivative term tuning matrix
<R>	<TuningPID>	Type, Size, Value	Tag for tuning matrix R used in LQR computations.
<Q>	<TuningPID>	Type, Size, Value	Tag for tuning matrix Q used in LQR computations.
<KS>	<TuningSliding>	Type, Size, Value	Tag for tuning matrix in Sliding Mode.
<KPID>	<TuningSliding>	Type, Size, Value	Tag for PID tuning matrix used in Sliding Mode controller.
<Lambda>	<TuningSliding>	Value	Tag for tuning in Sliding Mode
<Phi>	<TuningSliding>	Value	Tag for tuning in Sliding Mode
<IntegralLimit>	<TuningSliding>	Value	Tag for integral limits set on the integrator
<SensorInstalled>	<Vessel>	NumberOfSensors, SensorIDs	Tag used to set which sensors are used by the vessel.
<Thruster>	<Vessel>	ID, Name, Type, PosX, PosY, PosZ, RotTheta, RotPhi, MaxRPM, Coefficient	Tag for Thruster data.