DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

1222·2022 800 ANNI

UNIVERSITÀ DEGLI STUDI DI PADOVA

SIGNET

signet.dei.unipd.it

WOSS – World Ocean Simulation System - tcl

Padova (Italy)

30th of January – 3rd of February 2023

Federico Guerra, Filippo Campagnaro, Michele Zorzi

woss@guerra-tlc.com, filippo.campagnaro@unipd.it

WOSS – TCL examples

We have several WOSS examples provided with

- WOSS standalone: `/<woss_src>/samples`
- DESERT Framework: `/<desert_src>/DESERT_Framework/DESERT/samples/desert_with_woss_samples`

DESERT relevant ones:

- **`test_desert_woss_no_dbs.tcl`**
  WOSS channel model and PHY layer with custom sediment and bathymetry without use of GEBCO DBs. SSP is provided via txt files.
- **`test_desert_woss_dbs.tcl`**
  WOSS channel model with use of DBs for SSP and sediment and GEBCO for bathymetry.
- **`test_woss_waypoints_time_evo.tcl`**
  Same as `test_desert_woss_no_dbs.tcl` but with Waypoint WOSS mobility model and time evolution active.

WOSS – TCL - foundation classes

```
#setup the random generator stream
WOSS/Definitions/RandomGenerator/NS2 set rep_number_
$opt(rep_num)
#WOSS/Definitions/RandomGenerator/C   set seed_ $opt(rep_num)

#create the foundation classes prototypes
set ssp_creator          [new "WOSS/Definitions/SSP"]
set sediment_creator     [new "WOSS/Definitions/Sediment"]
set pressure_creator     [new "WOSS/Definitions/Pressure"]
set time_arr_creator     [new "WOSS/Definitions/TimeArr"]
```

WOSS – TCL - foundation classes

```
set time_reference       [new
"WOSS/Definitions/TimeReference/NS2"]

set transducer_creator   [new "WOSS/Definitions/Transducer"]

set altimetry_creator    [new
"WOSS/Definitions/Altimetry/Bretschneider"]

set rand_generator       [new
"WOSS/Definitions/RandomGenerator/NS2"]

#set rand_generator       [new
"WOSS/Definitions/RandomGenerator/C"]

#intialize the random generator

$rand_generator initialize
```

WOSS – TCL – foundation classes

```
#create the woss::DefHandler singleton
set def_handler [new "WOSS/Definitions/Handler"]
#plug the prototypes into the DefHandler
$def_handler setSSPCreator          $ssp_creator
$def_handler setSedimentCreator     $sediment_creator
$def_handler setPressureCreator     $pressure_creator
$def_handler setTimeArrCreator      $time_arr_creator
$def_handler setTransducerCreator   $transducer_creator
$def_handler setTimeReference       $time_reference
$def_handler setRandomGenerator     $rand_generator
$def_handler setAltimetryCreator    $altimetry_creator
```

WOSS – TCL – dbs configuration

```
#create the optional Result Db in either ASCII or binary
format. Configure space sampling in meters if you wish to
reduce the complexity. Two channel simulations will be
considered equal if their distance <= space_sampling

WOSS/Creator/Database/Textual/Results/TimeArr set
space_sampling 0.0

set db_res_arr [new
"WOSS/Creator/Database/Textual/Results/TimeArr"]

$db_res_arr setDbPathName
"${opt(db_res_path)}/test_aloha_with_dbs_res_arr.txt"
```

WOSS – TCL – dbs configuration

```
#binary format

#WOSS/Creator/Database/Binary/Results/TimeArr set
space_sampling 0.0

#set db_res_arr [new
"WOSS/Creator/Database/Binary/Results/TimeArr"]

#$db_res_arr setDbPathName
"${opt(db_res_path)}/test_aloha_with_dbs_res_arr.dat"
```

WOSS – TCL – dbs configuration

```
#create the optional DECK41 database and configure the paths

set db_sedim [new
"WOSS/Creator/Database/NetCDF/Sediment/DECK41"]

$db_sedim setDeck41DbTypeV2

$db_sedim setUpDeck41CoordinatesDb
"${opt(db_path)}/seafloor_sediment/DECK41_V2_coordinates.nc"

$db_sedim setUpDeck41MarsdenDb
"${opt(db_path)}/seafloor_sediment/DECK41_V2_marsden_square.n
c"

$db_sedim setUpDeck41MarsdenOneDb
"${opt(db_path)}/seafloor_sediment/DECK41_V2_marsden_one_degr
ee.nc"
```

WOSS – TCL – dbs configuration

```
#create the optional SSP netcdf db

set db_ssp [new
"WOSS/Creator/Database/NetCDF/SSP/WOA2013/MonthlyAverage"]

$db_ssp setDbPathName
"${opt(db_path)}/ssp/WOA2018/WOA2018_SSP_June.nc"

#create the optional bathymetry database and configure the
path

set db_bathy [new
"WOSS/Creator/Database/NetCDF/Bathymetry/GEBCO"]

$db_bathy setDbPathName
"${opt(db_path_gebco)}/bathymetry/GEBCO_2020.nc"

#configure the GEBCO database format of the input NetCDF db

$db_bathy use2DFifteenSecondsPrecision
```

WOSS – TCL – dbs configuration

```
#create the optional custom altimetry (flat altimetry is
automatically used by the lib)

#time evolution off

WOSS/Definitions/Altimetry/Bretschneider set
evolution_time_quantum   -1

#automatic range computations

WOSS/Definitions/Altimetry/Bretschneider set range
-1

#automatic range steps

WOSS/Definitions/Altimetry/Bretschneider set
total_range_steps        -1
```

WOSS – TCL – dbs configuration

```
#model parameters

WOSS/Definitions/Altimetry/Bretschneider set
characteristic_height    1.5

WOSS/Definitions/Altimetry/Bretschneider set average_period
3.0

set cust_altimetry   [new
"WOSS/Definitions/Altimetry/Bretschneider"]

#create the woss::WossDbManager

set db_manager [new "WOSS/Database/Manager"]

#plug the customer altimetry if required otherwise flat will
be used

#$db_manager setCustomAltimetry  $cust_altimetry
```

WOSS – TCL - custom dbs configuration

```
#set a custom sediment, altimetry and SSP for ALL channel computations involved.

set db_manager [new "WOSS/Database/Manager"]

$db_manager setCustomSediment   "Test Sedim" 1560 200 1.5 0.9 0.8 1.0

$db_manager setCustomAltimetry  $cust_altimetry

$db_manager setCustomSSP        0 "./ssp-test.txt"
```

WOSS – TCL - custom dbs configuration

```
#create a custom bathymetry: it is a line that starts at ($opt(start_lat), $opt(start_long)), it is valid for all bearings, and has four range/depth points. WossDbManager will provide bathymetry for (lat, long) points selecting the closest point from its custom bathymetry

$db_manager setCustomBathymetry $opt(start_lat) $opt(start_long) -500.0 4 0.0 100.0 500.0 200.0 1500.0 200.0 2500.0 100.0
```

WOSS – TCL – custom bathymetry

```
#we create a bathymetry grid 2500m x 2500m with depth
increasing from 40m to 240m

proc createBathymetryMap { } {

  global db_manager woss_utilities opt rbathy

  set curr_lon [ $woss_utilities getLongfromDistBearing
$opt(start_lat) $opt(start_long) -90.0 500.0 ]

  set bathy_incr [ expr (240.0 - 40.0) / 250.0 ]

  for { set bid 0.0 } { $bid < 2500.0 } { set bid [expr $bid
+ 10.0]} {
```

WOSS – TCL – custom bathymetry

```
    set curr_lat [ $woss_utilities getLatfromDistBearing
$opt(start_lat) $curr_lon 180.0 $bid ]

    for { set bid2 0.0 } { $bid2 < 250.0 } { set bid2 [expr
$bid2 + 10.0] } {

      #take random number from random generator

      set rb [$rbathy value]

      #start grid with PI/2 bearing (3 o'clock)

      #one (range, depth) pair at a time

      $db_manager setCustomBathymetry $lat $opt(start_long)
90.0 1 [expr 10.0 * $bid2] [expr $bathy_start + $bid2 *
$bathy_incr + $rb ]

    }

  }

}
```

WOSS – TCL – Bellhop creator

```
#should the temporary directory be automatically removed at
the end of the sim?
WOSS/Creator/Bellhop set woss_clean_workdir            1.0
#configure the latest bellhop time arrival result file syntax
WOSS/Creator/Bellhop set bellhop_arr_syntax            2
#enable time evolution when required
WOSS/Creator/Bellhop set evolution_time_quantum       -1.0
#number of total bellhop runs for each simulation
WOSS/Creator/Bellhop set total_runs                    5
```

WOSS – TCL – Bellhop creator

```
#configure the frequency step of each simulation, leave 0 for automatic config
WOSS/Creator/Bellhop set frequency_step                 0.0
#bellhop specific configuration
WOSS/Creator/Bellhop set total_range_steps              3000.0
WOSS/Creator/Bellhop set tx_min_depth_offset            0.0
WOSS/Creator/Bellhop set tx_max_depth_offset            0.0
WOSS/Creator/Bellhop set total_transmitters             1
```

WOSS – TCL – Bellhop creator

```
#bellhop specific configuration
WOSS/Creator/Bellhop set total_rx_depths                2
WOSS/Creator/Bellhop set rx_min_depth_offset           -0.1
WOSS/Creator/Bellhop set rx_max_depth_offset           0.1
WOSS/Creator/Bellhop set total_rx_ranges               2
WOSS/Creator/Bellhop set rx_min_range_offset           -0.1
WOSS/Creator/Bellhop set rx_max_range_offset           0.1
```

WOSS – TCL – Bellhop creator

```
#let bellhop decide the number of rays for each sim
WOSS/Creator/Bellhop set total_rays                    0.0
#ray fan angles
WOSS/Creator/Bellhop set min_angle                     -45.0
WOSS/Creator/Bellhop set max_angle                     45.0
#ssp precision and number of depth steps, reduce these to
simplify the sim
WOSS/Creator/Bellhop set ssp_depth_precision           1.0e-8
WOSS/Creator/Bellhop set normalized_ssp_depth_steps    100000
```

WOSS – TCL – Bellhop creator

```
#create the woss::BellhopCreator
set woss_creator [new "WOSS/Creator/Bellhop"]
#configure the dir for temporary simulations file results
$woss_creator setWorkDirPath "/dev/shm/woss/aloha_with_dbs/"
#configure the optional bellhop.exe path
$woss_creator setBellhopPath        ""
```

WOSS – TCL – Bellhop creator

```
#configure bellhop specific parameters
$woss_creator setBellhopMode          0 0 "A"
$woss_creator setBeamOptions          0 0 "B"
$woss_creator setBathymetryType       0 0 "L"
$woss_creator setBathymetryMethod     0 0 "S"
$woss_creator setAltimetryType        0 0 "L"
#set the simulation start and end time for all channels, used
only if time evolution is active
$woss_creator setSimulationTimes      0 0 1 1 2010 0 0 1 2 1
2010 0 0 1
```

WOSS – TCL – WossManager

```
#create the woss::WossManager in either single threaded
#WOSS/Manager/Simple set debug                          0
#WOSS/Manager/Simple set is_time_evolution_active -1.0
#WOSS/Manager/Simple set space_sampling                 0.0
#set woss_manager [new "WOSS/Manager/Simple"]
```

WOSS – TCL – WossManager

```
#or multi-threaded fashion
WOSS/Manager/Simple/MultiThread set debug                0
WOSS/Manager/Simple/MultiThread set is_time_evolution_active
-1.0
WOSS/Manager/Simple/MultiThread set concurrent_threads   0
WOSS/Manager/Simple/MultiThread set space_sampling       0.0
set woss_manager [new "WOSS/Manager/Simple/MultiThread"]
```

WOSS – TCL – Transducer Handler

```
#create the transducer handler

set transd_handler [new WOSS/Definitions/TransducerHandler]

#import an optional transducer link it to "ITC-3001" tag

$transd_handler importAscii "ITC-3001"
"$opt(db_path)/transducers/ITC/ITC-ITC-3001-17.5kHz.txt"

#during node creation the transducer tag should be linked via
positon object to WossCreator

$woss_creator setCustomTransducerType $position(i) 0 "ITC-
3001" 0.0 0.0 1.0 0.0
```

WOSS – TCL – Transducer Handler

```
#and also to the node's PHY layer along with the freq min / max range

$phy(i) setTransducerType  [expr [$data_mask getFreq] - [$data_mask getBandwidth] / 2.0 ] [expr [$data_mask getFreq] + [$data_mask getBandwidth] / 2.0 ] "ITC-3001"
```

WOSS – TCL – WossController

```
#create the woss::WossController and plug any optional db and
the mandatory object instances
set woss_controller [new "WOSS/Controller"]
$woss_controller setBathymetryDbCreator      $db_bathy #opt.
$woss_controller setSedimentDbCreator        $db_sedim #opt.
$woss_controller setSSPDbCreator             $db_ssp #opt.
$woss_controller setTimeArrResultsDbCreator $db_res_arr #opt.
```

WOSS – TCL – WossController

```
$woss_controller setWossDbManager      $db_manager #mand.
$woss_controller setWossManager        $woss_manager #mand.
$woss_controller setWossCreator        $woss_creator #mand.
$woss_controller setTransducerhandler $transd_handler #mand.

#call initialize
$woss_controller initialize
```

WOSS – TCL – WOSSphy lib

```
#detection thres in db
WOSS/Module/Channel set channel_eq_snr_threshold_db_  0
#symb. resolution time[s]
WOSS/Module/Channel set channel_symbol_resolution_    5e-3
#-1 means disabled otherwise time in s, remainder of the taps
outside this windows will be cropped
WOSS/Module/Channel set channel_eq_time_               -1
WOSS/Module/Channel set debug_                        0
#noise model related
WOSS/Module/Channel set windspeed_                    0.0
```

WOSS – TCL – WOSSphy lib

```
#noise model
WOSS/Module/Channel set shipping_                    0.0
#noise model
WOSS/Module/Channel set practical_spreading_         1.75
WOSS/Module/Channel set prop_speed_                  1500.0
#create the channel and add the channel estimator
set channel [new "WOSS/Module/Channel"]
$channel setWossManager      $woss_manager
#receives CLMSG for ch estimation between nodes. It keeps
track of the channel between two nodes
$channel setChannelEstimator $channel_estimator
```

WOSS – TCL – WOSSphy lib

```
#create the woss Mpropagation and link the woss::WossManager
set propagation [new "WOSS/MPropagation"]
$propagation setWossManager $woss_manager

#configure all BPSK classes
WOSS/Module/MPhy/BPSK  set AcquisitionThreshold_dB_   10.0
WOSS/Module/MPhy/BPSK  set BitRate_                $opt(bitrate)
WOSS/Module/MPhy/BPSK  set MaxTxSPL_dB_               190
WOSS/Module/MPhy/BPSK  set MinTxSPL_dB_               10
```

WOSS – TCL – WOSSphy lib

```
WOSS/Module/MPhy/BPSK  set MaxTxRange_                10000
WOSS/Module/MPhy/BPSK  set PER_target_                0.01
WOSS/Module/MPhy/BPSK  set TxSPLMargin_dB_            10
WOSS/Module/MPhy/BPSK  set RxSnrPenalty_dB_           -10.0
#use transducer to compute max SPL instead of MaxTxSPL_dB_,
and use channel estimation object to compute SPL required for
PER_target_
WOSS/Module/MPhy/BPSK  set SPLOptimization_           1
```

WOSS – TCL – Time evolution

```
#each object will evolve only if 100 seconds has passed
WOSS/Definitions/Altimetry/Bretschneider set
evolution_time_quantum   100


#configure the evolution time quantum in seconds for
woss::BellhopWoss objects
WOSS/Creator/Bellhop set evolution_time_quantum        3600.0
```

WOSS – TCL – Time evolution

```
#set different SSP for different Time values, to account for
time evolution
set time_evo_ssp_1 [new "WOSS/Definitions/Time"]
set time_evo_ssp_2 [new "WOSS/Definitions/Time"]
set time_evo_ssp_3 [new "WOSS/Definitions/Time"]

#first ssp, time key is 1st january 2011, 9:01 am
$time_evo_ssp_1 setTime 1 1 2011 9 0 1
#second ssp, time key is 1st january 2011, 12:01 am
$time_evo_ssp_2 setTime 1 1 2011 12 0 1
#third ssp, time key is 1st january 2011, 18:01 am
$time_evo_ssp_3 setTime 1 1 2011 18 0 1
```

WOSS – TCL – Time evolution

```
set db_manager [new "WOSS/Database/Manager"]
#insert the Altimetry with time evo active
$db_manager setCustomAltimetry  $cust_altimetry

#insert the custom SSPs, each SSP value with its related Time key
$db_manager setCustomSSP   $time_evo_ssp_1 "./ssp-test.txt"
$db_manager setCustomSSP   $time_evo_ssp_2 "./ssp-test_2.txt"
$db_manager setCustomSSP   $time_evo_ssp_3 "./ssp-test_3.txt"
```