

MOOS-IvP : Outils et visualisation

Simon Rohou

Décembre 2018

V1.01

Supports de cours disponibles sur

www.simon-rohou.fr/cours/moos-ivp/

Retour sur l'exercice du TP 1

Introduction

Outils complémentaires

Visualisation

Rejeu de mission

Voir aussi

Références

Section 1

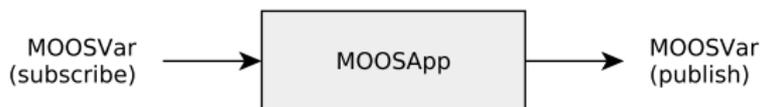
Retour sur l'exercice du TP 1

Retour sur l'exercice du TP 1

Une application MOOS : résumé

Interface d'une application MOOS :

- ▶ une **MOOSApp** réalise une tâche particulière
- ▶ elle a son propre processus
- ▶ elle communique avec d'autres **MOOSApp** *via* la MOOSDB à travers des variables **MOOSVar**
- ▶ cette communication se fait au sein d'une architecture *publish-subscribe* :
 - ▶ elle s'abonne à des **MOOSVar** (*subscribe*)
 - ▶ elle publie des **MOOSVar** (*publish*)

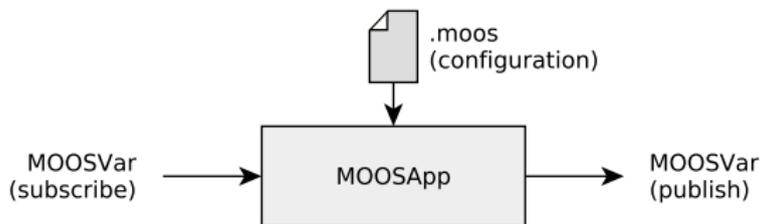


Retour sur l'exercice du TP 1

Une application MOOS : résumé

Paramétrage d'une application MOOS :

- ▶ une **MOOSApp** peut-être configurée *via* un fichier `.moos`
- ▶ les informations paramétrées dans ce fichier sont propres à cette application

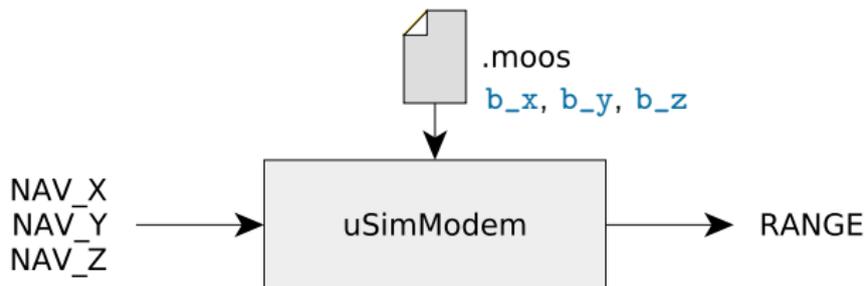


Retour sur l'exercice du TP 1

uSimModem

Pour l'application **uSimModem** :

- ▶ abonnements : **NAV_X**, **NAV_Y**, **NAV_Z** (la position du robot)
- ▶ publications : **RANGE**
- ▶ configurations : **b_x**, **b_y**, **b_z** (la position de la balise)



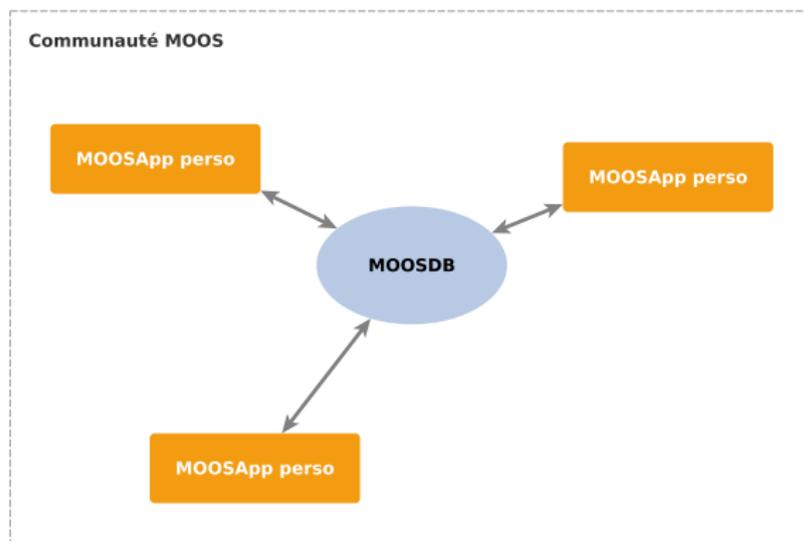
Section 2

Introduction

Introduction

Modularité d'une communauté MOOS

L'ajout d'une nouvelle **MOOSApp** se fait simplement en complétant le fichier `.moos` de la mission.

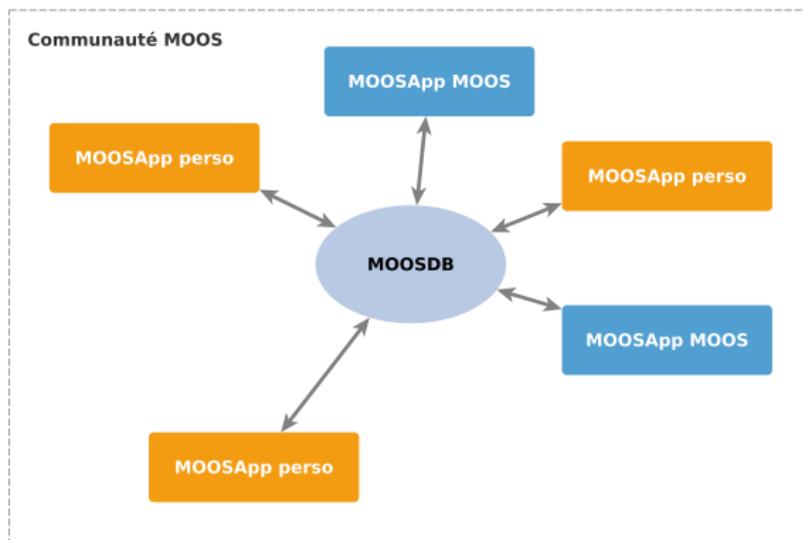


L'objectif de ce cours est de découvrir quelques **MOOSApp** déjà existantes, proposées par MOOS et le package IvP.

Introduction

Modularité d'une communauté MOOS

L'ajout d'une nouvelle **MOOSApp** se fait simplement en complétant le fichier `.moos` de la mission.

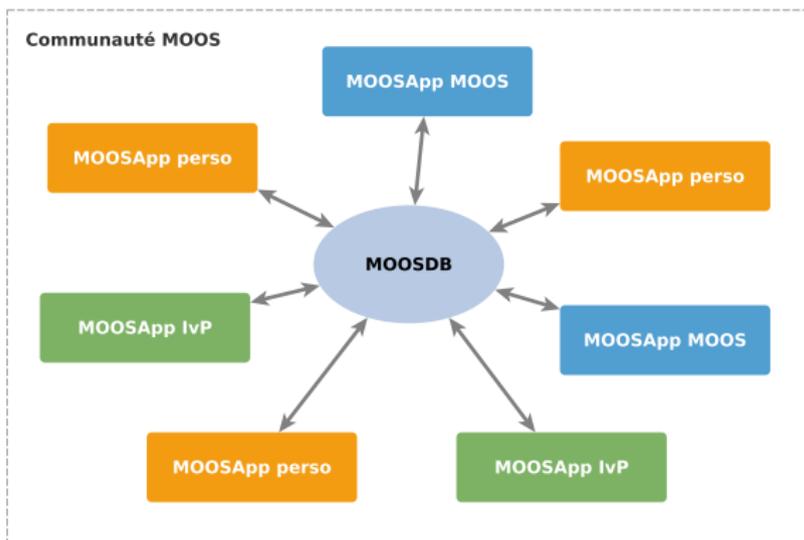


L'objectif de ce cours est de découvrir quelques **MOOSApp** déjà existantes, proposées par MOOS et le package lvP.

Introduction

Modularité d'une communauté MOOS

L'ajout d'une nouvelle **MOOSApp** se fait simplement en complétant le fichier `.moos` de la mission.



L'objectif de ce cours est de découvrir quelques **MOOSApp** déjà existantes, proposées par MOOS et le package IvP.

Introduction

La toolbox de MOOS-IvP

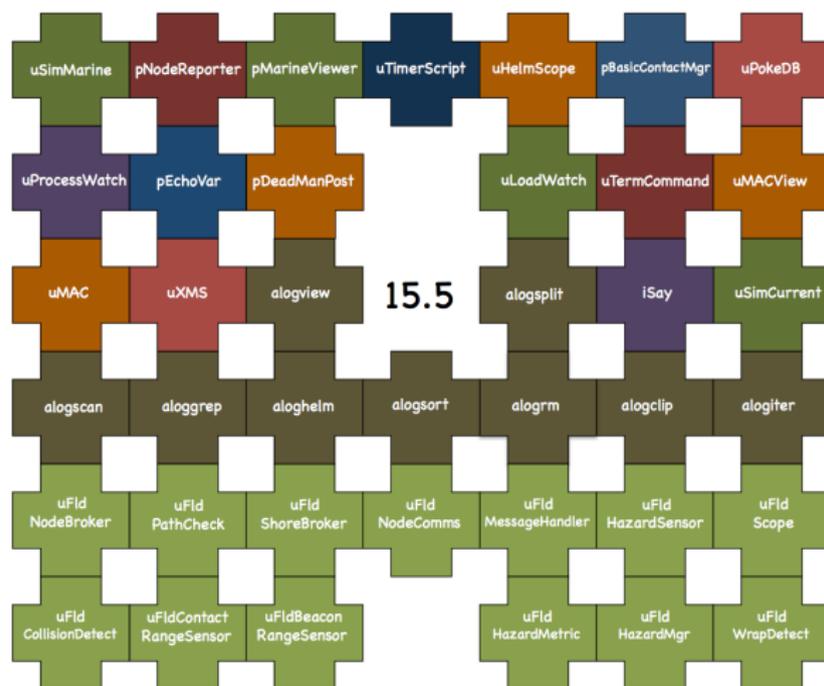


Figure – L'ensemble des MOOSApp proposées par IvP 15.5

Accès à la documentation des MOOSApp

► Documentation rapide :

Dans un terminal, entrer l'une des lignes de commande :

```
> pMoosApplication -h  
> pMoosApplication -e  
> pMoosApplication -i
```

- -h (--help) : donne le synopsis de l'application
- -e (--example) : donne un exemple de configuration .moos
- -i (--interface) : résume les publications et abonnements

► Documentation complète :

Voir la documentation en ligne de MOOS-IvP sur

<http://oceanai.mit.edu/ivpman/pmwiki/pmwiki.php?n=Tools.Cover>

Section 3

Outils complémentaires

Outils complémentaires

Découverte de quelques applications usuelles

Fonctions usuelles :

- ▶ `uTimerScript` : planifier des écritures dans la MOOSDB
- ▶ `pEchoVar` : dupliquer / renommer des variables MOOS
- ▶ `pNodeReporter` : faire une synthèse de l'état d'un robot

Applications en robotique marine :

- ▶ `uSimMarine` : simulation d'un robot marin
- ▶ `pMarineViewer` : affichage d'une carte, interface de contrôle

Enregistrement :

- ▶ `pLogger` : enregistrer les données d'une mission
- ▶ `uPlayback` : rejouer une mission

Outils complémentaires

uTimerScript : planifier des écritures dans la MOOSDB

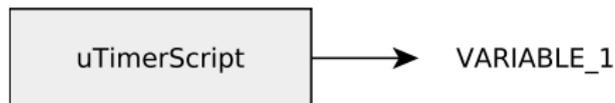
Rappel :

Pour publier manuellement dans la MOOSDB : **uPokeDB**
Ici, le processus est automatisé.

uTimerScript : exemple

Publication d'une variable **VARIABLE_1** (bool) :

```
1 ProcessConfig = uTimerScript
2 {
3   event = var=VARIABLE_1, val=true
4   event = quit
5 }
```

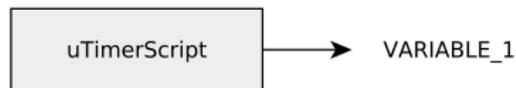


Outils complémentaires

uTimerScript : planifier des écritures dans la MOOSDB

uTimerScript : autre exemple

Publication d'une variable aléatoire toutes les 10s



```
1 ProcessConfig = uTimerScript
2 {
3   AppTick = 0.1 // Hz
4
5   // à chaque appel du script, un RANDOM_NUMBER est généré
6   randvar = varname=RANDOM_NUMBER, min=-100, max=100
7
8   // VARIABLE_1 prend ensuite la valeur de RANDOM_NUMBER
9   event = var=VARIABLE_1, val=${RANDOM_NUMBER}
10
11   reset_max = unlimited
12   reset_time = end
13 }
```

Outils complémentaires

pEchoVar : dupliquer / renommer des variables MOOS

pEchoVar : exemple

Publication de la valeur de `VARIABLE_1` sous un autre nom :



```
1 ProcessConfig = pEchoVar
2 {
3   AppTick = 20
4   CommsTick = 20
5
6   // réécriture de VARIABLE_1 dans VARIABLE_2
7   echo = VARIABLE_1 -> VARIABLE_2
8 }
```

Outils complémentaires

pEchoVar : dupliquer / renommer des variables MOOS

pEchoVar : autre exemple
Publication sous condition



```
1 ProcessConfig = pEchoVar
2 {
3   AppTick = 20
4   CommsTick = 20
5
6   // condition de re-publication
7   condition = VARIABLE_3 <= 67
8
9   // réécriture de VARIABLE_1 dans VARIABLE_2
10  echo = VARIABLE_1 -> VARIABLE_2
11 }
```

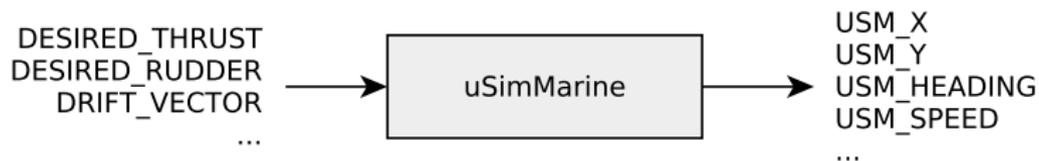
Outils complémentaires

uSimMarine : simulation d'un robot marin

Simulation d'équations d'état pré-écrites.

Application adaptée à un robot (sous-)marin.

- ▶ `DESIRED_THRUST` : commande envoyée au propulseur
- ▶ `DESIRED_RUDDER` : commande envoyée au servo-moteur
- ▶ `USM_X/Y/Z` : calcul de la position en $x/y/z$
- ▶ `USM_HEADING` : calcul du cap du robot



Outils complémentaires

uSimMarine : simulation d'un robot marin

Comportement du robot configurable selon divers critères. Ex :

- ▶ `max_acceleration` : accélération max
- ▶ `buoyancy_rate` : flottabilité (sous-marin)
- ▶ `thrust_map` : relation non linéaire entre thrust et vitesse

```
thrust_map = "-100:-3.5, -75:-3.2, -10:-2,  
20:2.4, 50:4.2, 80:4.8, 100:5"
```

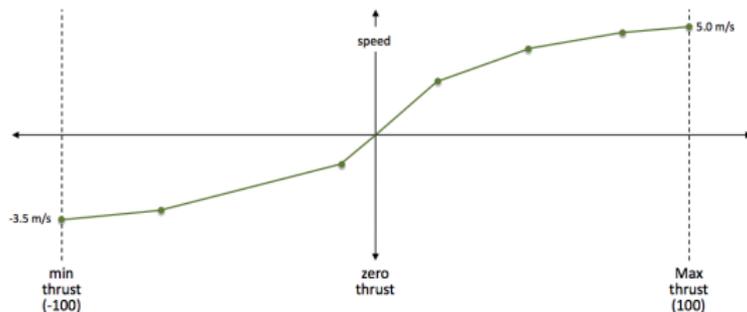


Figure – exemple de `thrust_map`

Outils complémentaires

uSimMarine : simulation d'un robot marin

Exemple de configuration ¹ :

```
1 ProcessConfig = uSimMarine
2 {
3     AppTick = 4
4     CommsTick = 4
5
6     start_pos = x=0, y=0, speed=0, heading=0, depth=0
7
8     drift_vector = 0,0 // heading, magnitude
9     buoyancy_rate = 0.025 // m/s
10    max_acceleration = 0 // m/s^2
11    max_deceleration = 0.5 // m/s^2
12    thrust_map = 0:0, 20:1, 40:2, 60:3, 80:5, 100:5
13
14    prefix = NAV // par défaut : USM
15 }
```

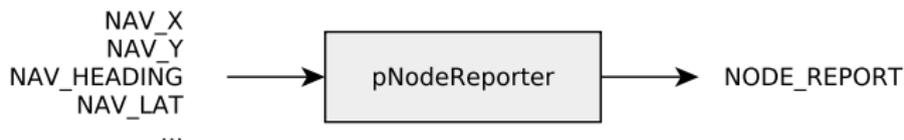
1. **Note** : il n'est pas nécessaire de redéfinir tous les paramètres.
Les paramètres non-définis prendront une valeur par défaut.

Outils complémentaires

pNodeReporter : faire une synthèse de l'état d'un robot

Objectif : rassembler dans une `MOOSVar` l'état d'un robot.

La variable produite, `NODE_REPORT`, sert de synthèse pour visualiser un robot. L'information peut ainsi facilement être propagée vers d'autres communautés.



Exemple :

```
NODE_REPORT = "NAME=alpha,TYPE=UUV,TIME=1252348077.59,  
X=51.71,Y=-35.50,  
LAT=43.824981,LON=-70.329755,  
SPD=2.00,HDG=118.85,YAW=118.84754,  
DEP=4.63,LENGTH=3.8"
```

Outils complémentaires

pNodeReporter : faire une synthèse de l'état d'un robot

Exemple de configuration :

```
1 ProcessConfig = pNodeReporter
2 {
3   // Caractéristiques du robot
4   platform_type    = glider // valeurs : {uuv,auv,ship,kayak}
5   platform_length  = 8      // m
6   ...
7 }
```

L'application prend son sens dans un contexte multi-robots :

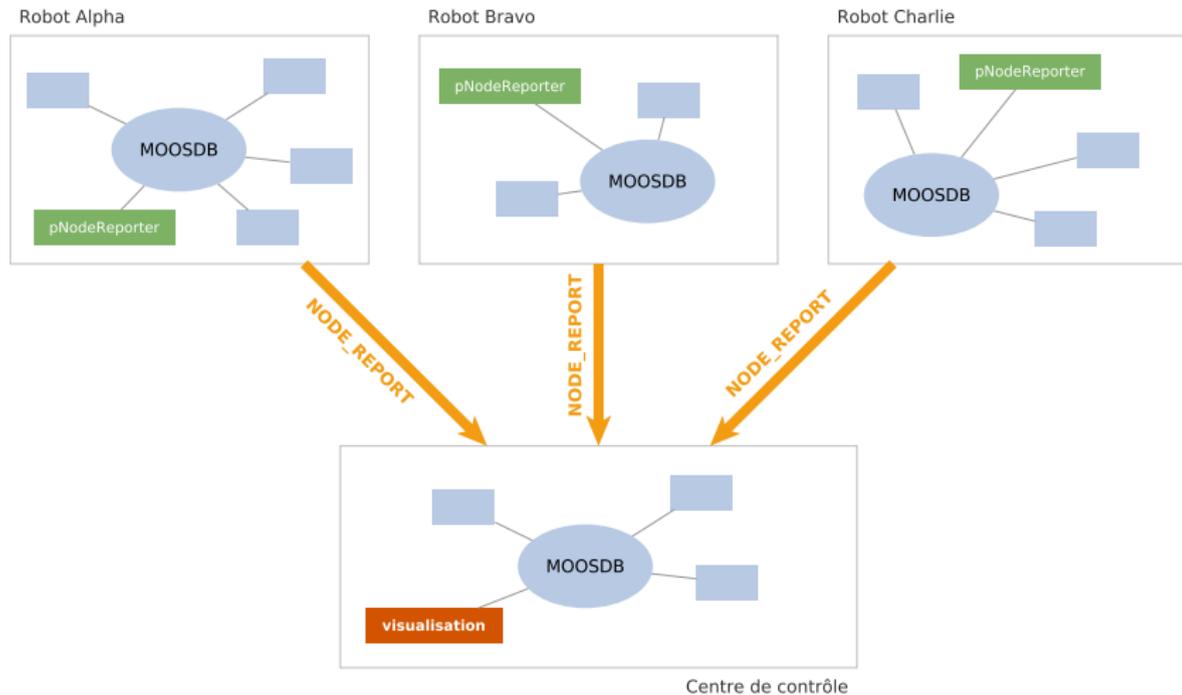
1 robot \iff 1 communauté MOOS \iff 1 `NODE_REPORT`

Une communauté dédiée à la visualisation peut ainsi recueillir les `NODE_REPORT` d'autres communautés et afficher un groupe de robots.

Outils complémentaires

pNodeReporter : utilisation en multi-communautés

Monitoring de 3 robots au sein d'une communauté de supervision :



Section 4

Visualisation

Visualisation

pMarineViewer : contrôle et visualisation

pMarineViewer (MIT Version 15.5) localhost

File BackView GeoAttr Vehicles AppCasting MOOS-Scope

Node	AC	CW	HW	App	AC	CW	HW
instaboat	0	0	0	pmocintfy	23	0	0
				MFLabelBroker	15	0	0
				pmodeReporter	15	0	0
				pmocintfy	15	0	0
				MScheduler_Toyentick	15	0	0
				IPeGlu	4243	0	0
				planning	15	0	0
				PLATLanLocalIrid	15	0	0
				plScheduler_Helix	15	0	0
				plProcessWatch	12	0	0
				lCamera1	3	0	0

IPNode instaboat: 0/0(44077)

Node's status: ok

Pin	Monitor	Value	Min	Max	Zero	Reset	Reversed	Bilateral
1	FDMAL_SELLIAGE_RUBBER	1580	1380	1580	2002	yes	yes	
0	FDMAL_SELLIAGE_TAPROT	1580	1580	1580	1380	no	no	

Pin	Monitor	Value	Threshold
1	BATTERY_OVERAGE	0.002 A	
0	BATTERY_VOLTAGE	3.942 V	3.588 V

VName: instaboat X(m): 34.0 Lat: 0.000000 Spd: 0.1 Dep(m): 0.0 Time: 1784.6 DEPLOY

VType: kayak Y(m): 54.6 Lon: 0.000000 Hdg: 91.2 Age(s): 0.00 Warp: 1 GO_HOME

Variable: jva Trn: jva Value: [To add Scope Variables: SCOPE=VARNAME in the MOOS config block]

Visualisation

pMarineViewer : contrôle et visualisation

pMarineViewer (MIT Version 15.5) localhost

File BackView GotoAttr Vehicles AppCasting MOOS-Scope

Node	AC	IX	SR	App	AC	IX	SR
motoplacement	0	0	0	motoplacement	23	0	0
motoplacement	0	0	0	MP_Calculateur	15	0	0
motoplacement	438	0	0	MP_MoteurPorteur	15	0	0
				motoplacement	12	0	0
				MOOSScope_Injecteur	14	0	0
				MP_GUI	4243	0	0
				MOOSScope	12	0	0
				MOOSScopeOperatorUI	15	0	0
				MOOSScope_Media	12	0	0
				MOOSScopeUI	12	0	0
				SCAManager	5	0	0

MOOS Scope

File	Node	Value	Units	Desc	Pos	Reverse	Scale	Label
1	170m_011002_00000	1188	130	1388	202	yes	m	
2	170m_011002_00000	1388	130	1388	202	no	m	

File

File	Node	Value	Units
1	BATTERY_000000	0.012	A
2	BATTERY_001002	1.000	V

Name: jetboat X(m): 14.0 Y(m): 14.6 Lat: 0.000000 Lon: 0.000000 Spd: 0.1 Hdg: 91.2 Dep(m): 0.0 Agt: 0.00 Time: 1784.6

Variable: jvc Title: jvc Value: [To edit Scope Variables: SCOPE=VARIABLE in the MOOS config block]

DEPLOY GO HOME

Visualisation

pMarineViewer : contrôle et visualisation

pMarineViewer (MIT Version 15.5) localhost

File BackView GeoAttr Vehicles AppCasting MOOS-Scope

Node	AC	CW	HW	App	AC	CW	HW
instaboat	4385	0	0	planning	21	0	0
				plNodeBroker	15	0	0
				plNodeReporter	15	0	0
				plNodeV	15	0	0
				plScheduler_ToyTrack	15	0	0
				IPeGrip	4243	0	0
				plSensor	15	0	0
				plPLanLocalGrid	15	0	0
				plScheduler_Helm	15	0	0
				plProcessWatch	12	0	0
				plCamera	3	0	0

plNode instaboat (N16140077)

Node's status: ok

Pin	Monitor	Value	Min	Max	Zero	Max	Reversed	Bilateral
1	FIND_SELECTED_RUBBER	1588	1381	1588	2812	yes	yes	
2	FIND_SELECTED_TURTLE	1588	1588	1588	1588	no	no	

Pin	Monitor	Value	Threshold
1	BATTERY_OVERAGE	8.022 A	
2	BATTERY_VOLTAGE	3.942 V	3.588 V

instaboat

toyTrack

toyTrack (3000, 16000, 200)

WName: instaboat X(m): 14.0 Lat: 0.000000 Spd: 0.1 Dep(m): 0.0 Time: 1784.6 DEPLOY

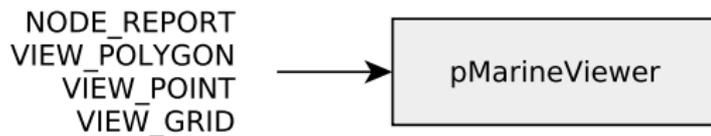
VType: kayak Y(m): 14.6 Lon: 0.000000 Hdg: 91.2 Age(s): 0.00 Warp: 1 GO_HOME

Variable: pos Tra: pos Value: To edit Scope Variables: SCOPE=VARIABLE in the MOOS config box

Visualisation

pMarineViewer : contrôle et visualisation

`NODE_REPORT` contient les informations pour afficher un véhicule.
D'autres variables telles que `VIEW_POLYGON` permettent d'afficher des éléments simples.



Exemple : la `MOOSVar` suivante permet d'afficher un triangle :

```
VIEW_MARKER = "type=triangle,lat=42.358,lon=-71.0874,  
scale=2,color=blue,width=8"
```

La publication de cette `MOOSVar` met à jour `pMarineViewer`.

Visualisation

pMarineViewer : contrôle et visualisation

The screenshot displays the pMarineViewer application interface. The main window shows a satellite map of a coastal area with a blue grid overlay. A yellow and green path is visible on the map, representing the movement of a vehicle. Two octagonal polygons are drawn on the map. The interface includes a menu bar at the top with options: File, BackView, GeoAttr, Vehicles, AppCasting, MOOS-Scope, ClearHistory, and Action. A dropdown menu is open under 'GeoAttr', showing options for Polygons, SegLists, Points, Vectors, Circles, Grids, Markers, RangePulses, CommsPulses, Datum, OpArea, and DropPoints. The 'Polygons' menu is expanded, showing radio buttons for 'polygon_viewable_all=true' (selected), 'polygon_viewable_all=false', and 'Toggle Polygons' (P). Below these are radio buttons for 'polygon_viewable_labels=true' and 'polygon_viewable_labels=false' (selected), along with a 'Toggle Polygon Labels' button (⇧P). At the bottom of the window, there is a control panel with various input fields and buttons. The fields contain the following values: VName: henry, X(m): -11.6, Lat: 43.824877, Spd: 1.2, Dep(m): 0.0, Time: 52.5, VType: kayak, Y(m): -46.7, Lon: -70.330534, Hdg: 197.7, Age(s): 0.45, Warp: 1, Variable: TESTING_PSHARE, Tm: (empty), and Value: (empty). Buttons for PERMUTE, DEPLOY, and RETURN are also present.

VName: henry	X(m): -11.6	Lat: 43.824877	Spd: 1.2	Dep(m): 0.0	Time: 52.5	PERMUTE	DEPLOY
VType: kayak	Y(m): -46.7	Lon: -70.330534	Hdg: 197.7	Age(s): 0.45	Warp: 1		RETURN
Variable: TESTING_PSHARE	Tm:	Value:					

Visualisation

pMarineViewer : contrôle et visualisation

pMarineViewer peut aussi servir d'interface de contrôle :
1 clic sur un bouton \implies 1 publication dans la MOOSDB



Pour ajouter des boutons dans l'interface :

```
1 ProcessConfig = pMarineViewer
2 {
3   bouton_one = Bouton1 # VARIABLE_1=35.2
4   bouton_two = Bouton2 # VARIABLE_2=true
5
6   // Syntaxe :
7   bouton_one = <label> # <MOOSVar>=<value> # <MOOSVar>=<val...
8 }
```

Visualisation

pMarineViewer : ajout d'un fond de carte

**Exemple** : affichage de moulin_blanc.tif

1. définir une origine dans le fichier .moos.

```
// En en-tête de fichier
```

```
LatOrigin = 48.715203
```

```
LongOrigin = 2.209163
```

2. configurer pMarineViewer :

```
ProcessConfig = pMarineViewer
```

```
{
```

```
  tiff_file = moulin_blanc.tif // chemin relatif
```

```
  zoom = 1.2
```

```
}
```

3. dans le répertoire de la mission, ajouter :

- ▶ moulin_blanc.tif : la carte (format TIFF)

- ▶ moulin_blanc.info : les coordonnées des coins de la carte :

```
  lat_north = 48.39972
```

```
  lat_south = 48.38361
```

```
  lon_east = -4.41222
```

```
  lon_west = -4.44166
```

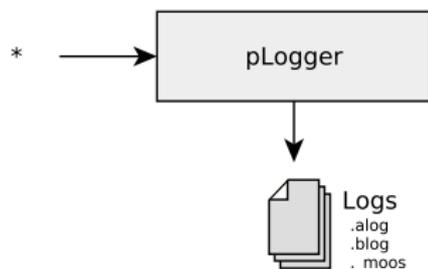
Section 5

Rejeu de mission

Rejeu de mission

pLogger : enregistrer les données d'une mission

Les changements des **MOOSVar** sont enregistrés dans des fichiers :



Des fichiers de logs sont créés pour chaque mission :

- ▶ `file.alog` : valeurs datées de chaque **MOOSVar**
- ▶ `file.blog` : logs de **MOOSVar** binaires
- ▶ `file._moos` : sauvegarde du fichier de mission `.moos` utilisé

Rejeu de mission

pLogger : exemple de configuration

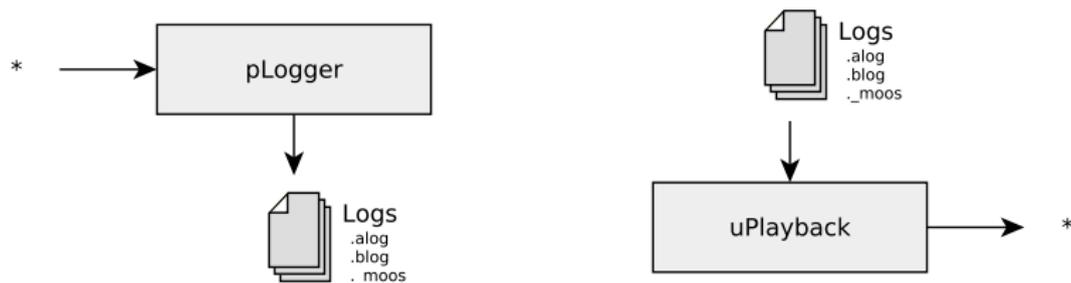
```
1 ProcessConfig = pLogger
2 {
3     // fichiers de logs préfixés par :
4     File = logsExempleMission
5
6     // emplacement du répertoire de logs
7     Path = /home/user/MOOSData/ExempleMission/
8
9     // MOOSVar à logger en particulier
10    // Log = MOOSVAR @ FREQUENCE
11    Log = CAMERA_IMAGE @ 1.0
12    Log = SONAR_DATA @ 0.5
13    Log = DESIRED_THRUST @ 0.1
14
15    // pour logger la totalité des changements dans la MOOSDB :
16    WildCardLogging = true
17
18    DoublePrecision = 12
19 }
```

Rejeu de mission

uPlayback : rejouer une mission

uPlayback utilise les logs de **pLogger**.

Les changements des **MOOSVar** sont reproduits dans la MOOSDB :



uPlayback relance la mission telle que configurée dans le `._moos`

Plus d'information :

<http://www.robots.ox.ac.uk/~pnewman/MOOSDocumentation/Essentials/>

[Logging/latex/pLogger.pdf](http://www.robots.ox.ac.uk/~pnewman/MOOSDocumentation/Essentials/Logging/latex/pLogger.pdf)

Section 6

Voir aussi

Voir aussi

Le reste du package IvP

De nombreuses autres applications sont disponibles sur la plate-forme de MOOS-IvP : <http://oceanai.mit.edu/ivpman/pmwiki/pmwiki.php?n=Tools.Cover>

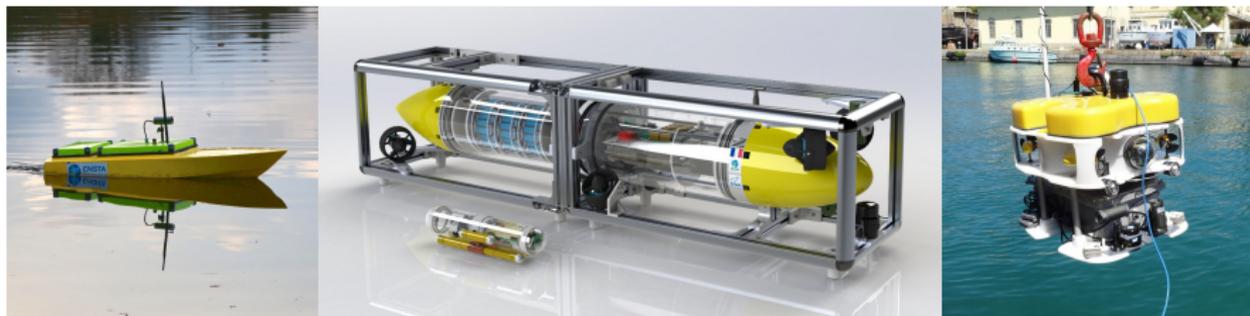
En particulier :

- ▶ **pShare** : communications entre communautés MOOS
- ▶ **uSimCurrent** : simulation de courants marins
- ▶ **pMarinePID** : régulation d'un robot marin

Voir aussi

MOOS-IvP à l'ENSTA Bretagne

MOOS-IvP est aussi utilisé sur les robots de l'ENSTA Bretagne :



Une **MOOSApp** a été développée pour chaque capteur de ces robots.

Le code est disponible sur Github :

www.github.com/ENSTABretagneRobotics/moos-ivp-enstabretagne

Références

- ▶ Page officielle de la V10 de MOOS (Oxford)
- ▶ Site officiel de MOOS-IvP
- ▶ Documentation en ligne de MOOS-IvP

Publication :

- M. Benjamin, H. Schmidt, P. Newman, J. Leonard,
Nested Autonomy for Unmanned Marine Vehicles with MOOS-IvP,
Journal of Field Robotics, Volume 27, Issue 6, pp. 834-875,
November 2010.

