NATO OTAN NURC
a NATO Research Centre
un Centre de Recherche de l'OTAN
NURC
PARTNERING FOR MARITIME INNOVATION

Reprint Series **NURC-PR-2010-002**

pOctaver, adding a scripting language to MOOS

Arjan Vermeij and Thomas J. Pastore

August 2010

Originally presented at:

MOOS Development and Applications Working Group – Cambridge, MA, 24-25 August 2010.

About NURC

Our vision

- To conduct maritime research and develop products in support of NATO's maritime operational and transformational requirements.
- To be the first port of call for NATO's maritime research needs through our own expertise, particularly in the undersea domain, and that of our many partners in research and technology.

One of three research and technology organisations in NATO, NURC conducts maritime research in support of NATO's operational and transformation requirements. Reporting to the Supreme Allied Commander, Transformation and under the guidance of the NATO Conference of National Armaments Directors and the NATO Military Committee, our focus is on the undersea domain and on solutions to maritime security problems.

The Scientific Committee of National Representatives, membership of which is open to all NATO nations, provides scientific guidance to NURC and the Supreme Allied Commander Transformation.

NURC is funded through NATO common funds and respond explicitly to NATO's common requirements. Our plans and operations are extensively and regularly reviewed by outside bodies including peer review of the science and technology, independent national expert oversight, review of proposed deliverables by military user authorities, and independent business process certification.

NOTE: The NURC Reprint series reprints papers and articles published by NURC authors in the open literature as an effort to widely disseminate NURC products. Users should cite the original article where possible.

pOctaver, adding a scripting language to MOOS

Arjan Vermeij and Thomas J. Pastore

This document, which describes work performed under Unmanned Surface Vehicles for Defence against Terrorism (EPOW 2009) has been approved by the Director.

Intentionally blank page

pOctaver, adding a scripting language to MOOS

Arjan Vermeij and Thomas J. Pastore

Executive Su mmary: NURC uses MOOS-IvP in a scientific research and rapid prototyping environment. Contributions to the open-source suite of MOOS-IvP software is readily made by programmers adept at C++ programming. However, the utility of MOOS-IvP's benefits can be expanded to additional users who normally develop their work in Matlab.

Octave is an open source Matlab clone. We developed pOctaver, which can directly run a script written in Octave as a MOOS application. This has great advantages in allowing for rapid prototyping. A complex function or new concept developed and preliminarily tested in Matlab can be quickly inserted into an existing moos mission for system-level simulation. The subsequent step from simulation in the MOOS environment to running on the vehicle follows immediately. The cycle from developing a concept and testing it in Matlab to running it on a vehicle can be reduced to hours or minutes, as contrasted with a timeline normally measured in days or longer if a conversion from Matlab code to C++ were required. pOctaver is a path by which a researcher can test a new function or application without requiring a single line of C-code to be written.

Of course there is always a place for well-written and computationally more efficient C++ programs, and Octave/Matlab can never compete with C++ if computation times are critical. The advantage of pOctaver from an organizational standpoint is that the efforts of the researchers and programmers can be decoupled and are no longer serially dependent prior to a first in-water test of a new concept.

The design and implementation of pOctaver is discussed, along with some examples of how it has been used at NURC in recent months.

This work was initially presented at MOOS-DAWG (MOOS Development and Applications Working Group) meeting, August 2010, Cambridge, Mass. USA This document consists of the abstract, the presentation slides from the working group meeting, and the pOctaver source files as an annex.

Intentionally blank page

pOctaver, adding a scripting language to MOOS

Arjan Vermeij and Thomas J. Pastore

Abstract: NURC uses MOOS-IvP in a scientific research and rapid prototyping environment. Contributions to the open-source suite of MOOS-IvP software is readily made by programmers adept at C++ programming. However, the utility of MOOS-IvP's benefits can be expanded to additional users who normally develop their work in Matlab.

Octave is an open source Matlab clone. We developed pOctaver, which can directly run a script written in Octave as a MOOS application. This has great advantages in allowing for rapid prototyping. A complex function or new concept developed and preliminarily tested in Matlab can be quickly inserted into an existing moos mission for system-level simulation. The subsequent step from simulation in the MOOS environment to running on the vehicle follows immediately. The cycle from developing a concept and testing it in Matlab to running it on a vehicle can be reduced to hours or minutes, as contrasted with a timeline normally measured in days or longer if a conversion from Matlab code to C++ were required. pOctaver is a path by which a researcher can test a new function or application without requiring a single line of C-code to be written.

Of course there is always a place for well-written and computationally more efficient C++ programs, and Octave/Matlab can never compete with C++ if computation times are critical. The advantage of pOctaver from an organizational standpoint is that the efforts of the researchers and programmers can be decoupled and are no longer serially dependent prior to a first in-water test of a new concept.

The design and implementation of pOctaver is discussed, along with some examples of how it has been used at NURC in recent months.

This work was initially presented at MOOS-DAWG (MOOS Development and Applications Working Group) meeting, August 2010, Cambridge, Mass. USA This document consists of the abstract, the presentation slides from the working group meeting, and the pOctaver source files as an annex. The annex is distributed as a tar file which is separate from the current pdf document, but integral to this reprint and may be requested by sending an email to pao@nurc.nato.int.

Keywords: Autonomy, computer programming, software design.

pOctaver, adding a scripting language to MOOS

Arjan Vermeij
Tom Pastore

August 25, 2010

Software is good ...

- it adds functionality
  - interfaces to external actuators/sensors
  - behaviours
  - simulation
- it's at the heart of every trial/experiment

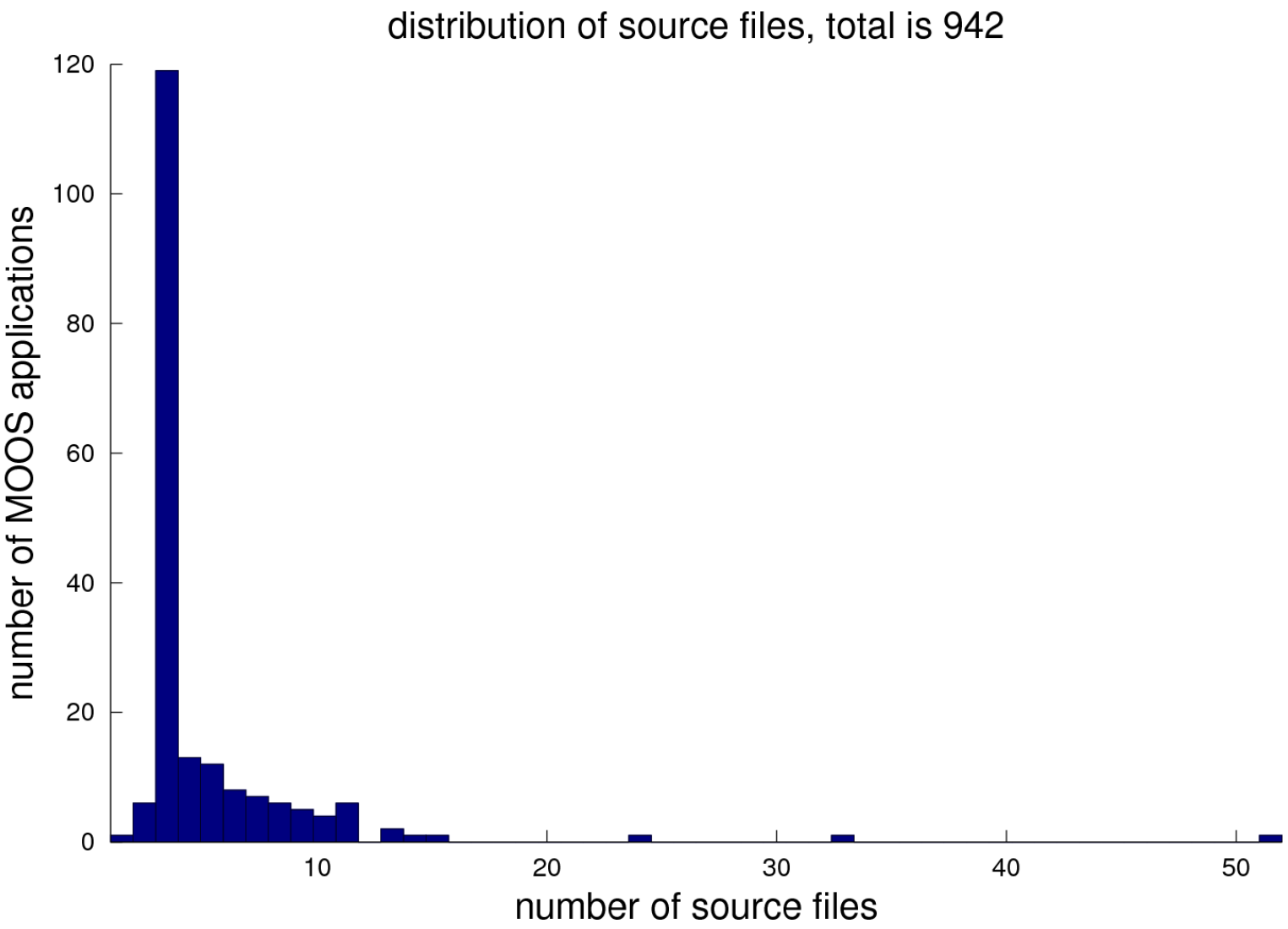
Software is bad ...

- it has to be
  - downloaded
  - compiled by everybody
  - maintained
  - rewritten
  - discarded
- it requires software engineering skills

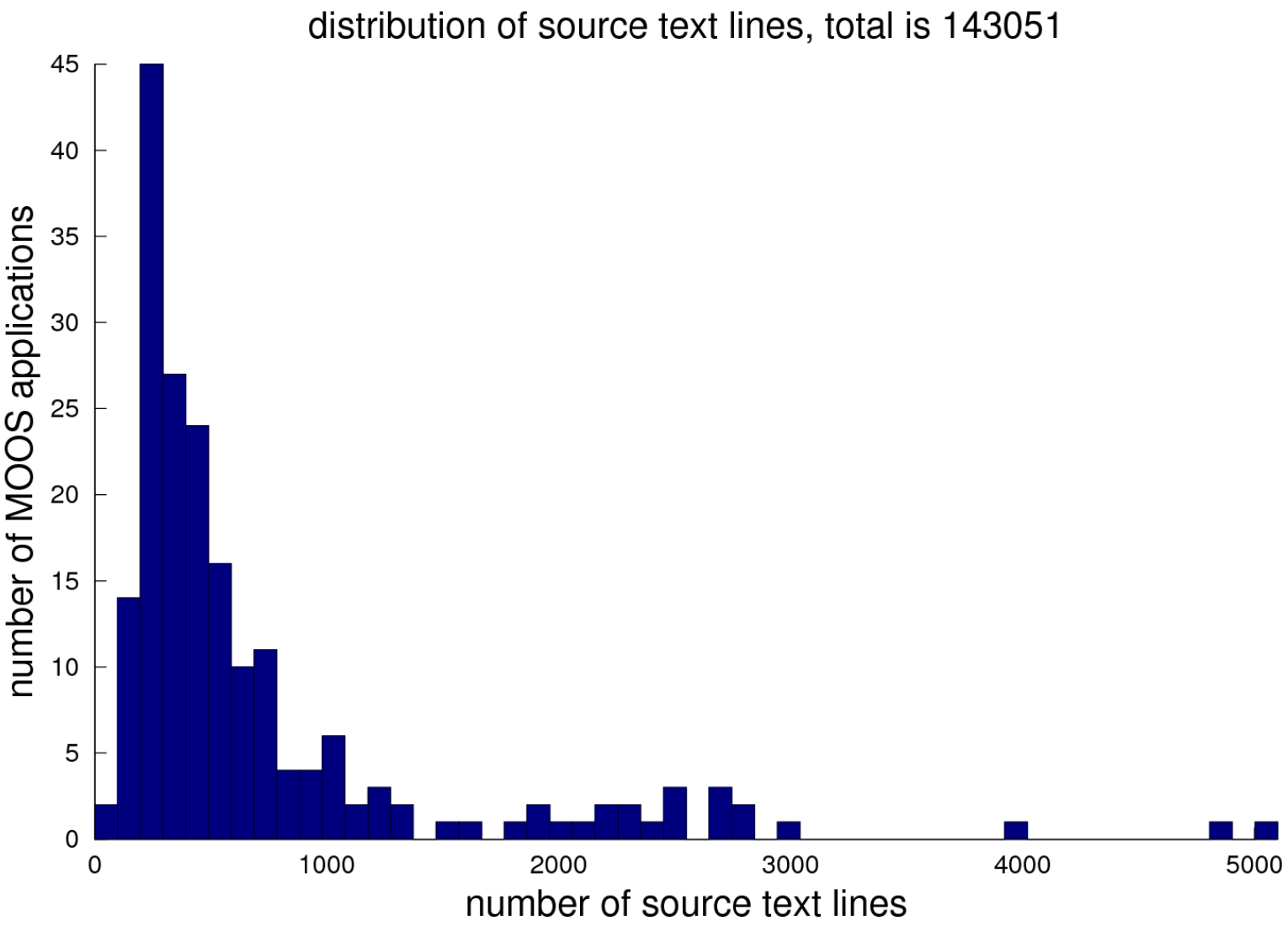
Software

- is a **means** to and end
- we want to have as little software as possible
- how much software do we have?

Source files



Source text lines



Two 'Solutions'

- NurcMoosApp
  - may reduce the amount of bookkeeping code
  - aims to help improve the quality of the code
- pOctaver
  - higher level scripting language

NurcMoosApp

- a subclass of MOOSApp
- has a somewhat simpler interface
- avoids some common pitfalls
- provides some functionality potentially useful to MOOS applications
  - initialisers
  - configurable variable names

NurcMoosApp — API

```
bool readMissionParameters (CProcessConfigReader&);
bool registerMoosVariables ();
void onNewMessage (const CMOOSMsg&);
bool Iterate ();

template <typename T> static T
getConfigParameter
    (const std::string name, const T defaultValue);
const VariableNames& variableNames () const;
```

NurcMoosApp

- a subclass of MOOSApp
- has a somewhat simpler interface
- avoids some common pitfalls
- provides some functionality potentially useful to MOOS applications
  - initialisers
  - configurable variable names

NurcMoosApp — Mission File

```
ProcessConfig = someNurcMoosApp
{
  initialiser.string = FAVOURITE_COLOUR = red
  initialiser.string = CPU_TEMPERATURE = 30

  variable-name.colour = FAVOURITE_COLOUR
  variable-name.temperature = TEMPERATURE
}
```

NurcMoosApp — Variable Names

```
m_Comms.Register (variableNames ().get ("colour"), 0);


if (moosMessage.GetKey ()
      == variableNames ().get ("colour"))
   ...

m_Comms.Notify (variableNames ().get ("colour"), ...
```

Octave

- an open source Matlab ® clone
- offers
  - matrices, strings, regular expressions
  - solving linear equations
  - solving nonlinear differential equations
  - plotting
- lacks
  - many advanced toolboxes

Example — negate

```
ProcessConfig = ANTLER
{
  Run = pOctaver @ NewConsole = false ~ pOctaver.negate
}

ProcessConfig = pOctaver.negate
{
  OctaveFunction = negate
  argument.in = negate.in
  argument.out = negate.out
}
```

Example — negate

negate.m

```
function negatedValue = negate (value)
   negatedValue = -value;
end
```

input/output

```
negate.in    negate.out
 3.00000      -3.00000
-5.00000       5.00000
```

Example — multiplier

```
ProcessConfig = pOctaver.multiplier
{
  OctaveFunction = multiplier
  argument.in = multiplier.factor
  argument.in = multiplier.in
  argument.out = multiplier.out
  initialiser.double = multiplier.factor = 3
}
```

Example — multiplier

multiplier.m

```
function result = multiplier (factor, value)
  result = (factor * value);
end
```

input/output

```
multiplier.factor  multiplier.in  multiplier.out
      3.00000         3.00000         9.00000
                     -5.00000       -15.00000
      4.00000                        -20.00000
```

Example — accumulator

accumulator.m

```
function result = accumulator (value)
  persistent accumulated = 0;
  accumulated += value;
  result = accumulated;
end
```

input/output

accumulator.in	accumulator.out
1.00000	1.00000
2.00000	3.00000
3.00000	6.00000

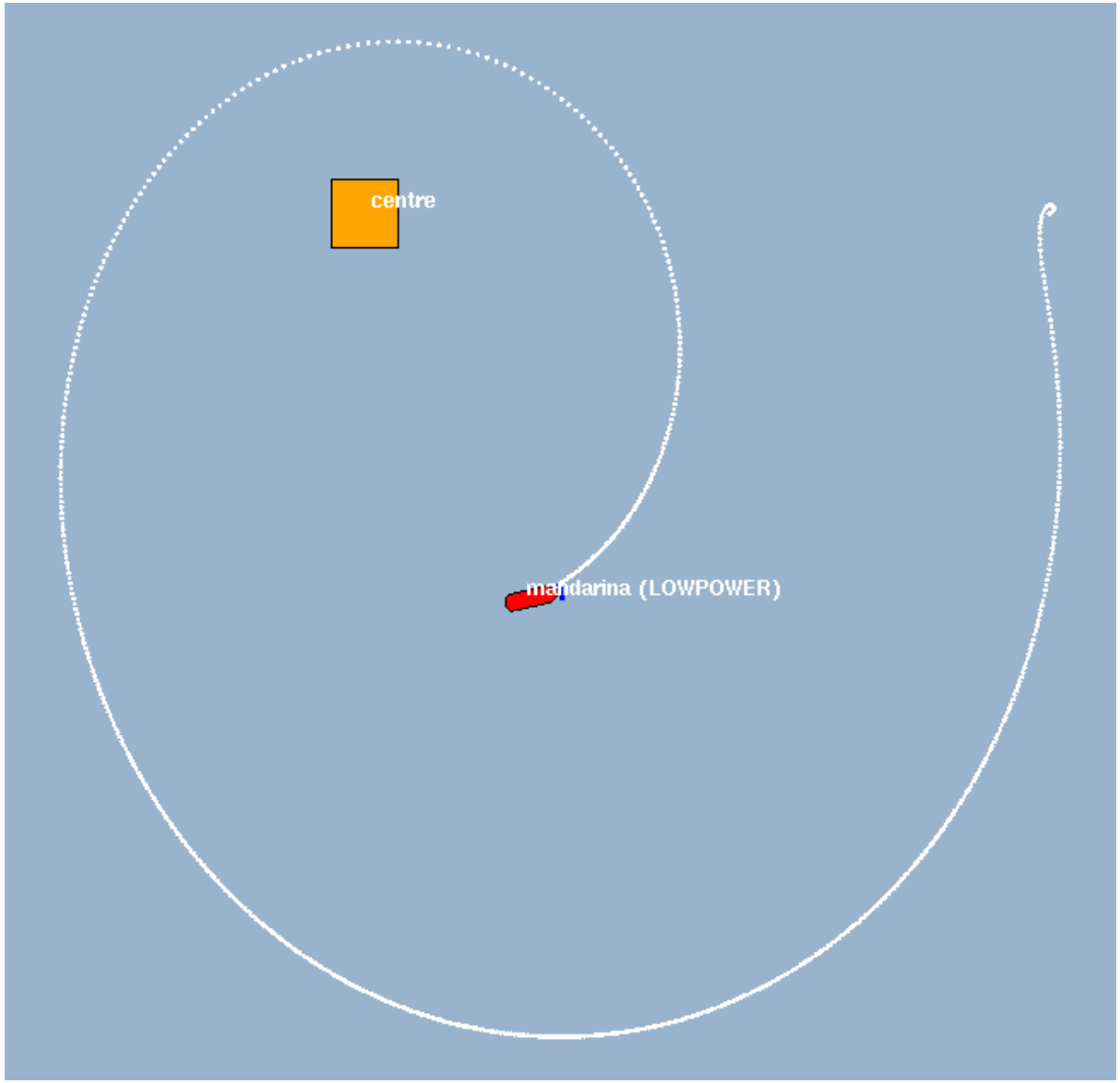
Example — behaviour

```
function headingUpdates = maintainRelativeBearing
  (relativeBearing, contactX, contactY, navX, navY)

  contactAngle

      = atan2 ((contactY - navY), (contactX - navX));

  desiredRelativeAngle = - (relativeBearing / 180.0 * pi);

  desiredVehicleAngle = (contactAngle - desiredRelativeAngle);

  heading = mod

      ((90.0 - (desiredVehicleAngle * 180.0 / pi)), 360.0);

  headingUpdates = sprintf ('heading = %.2f', heading);

end
```

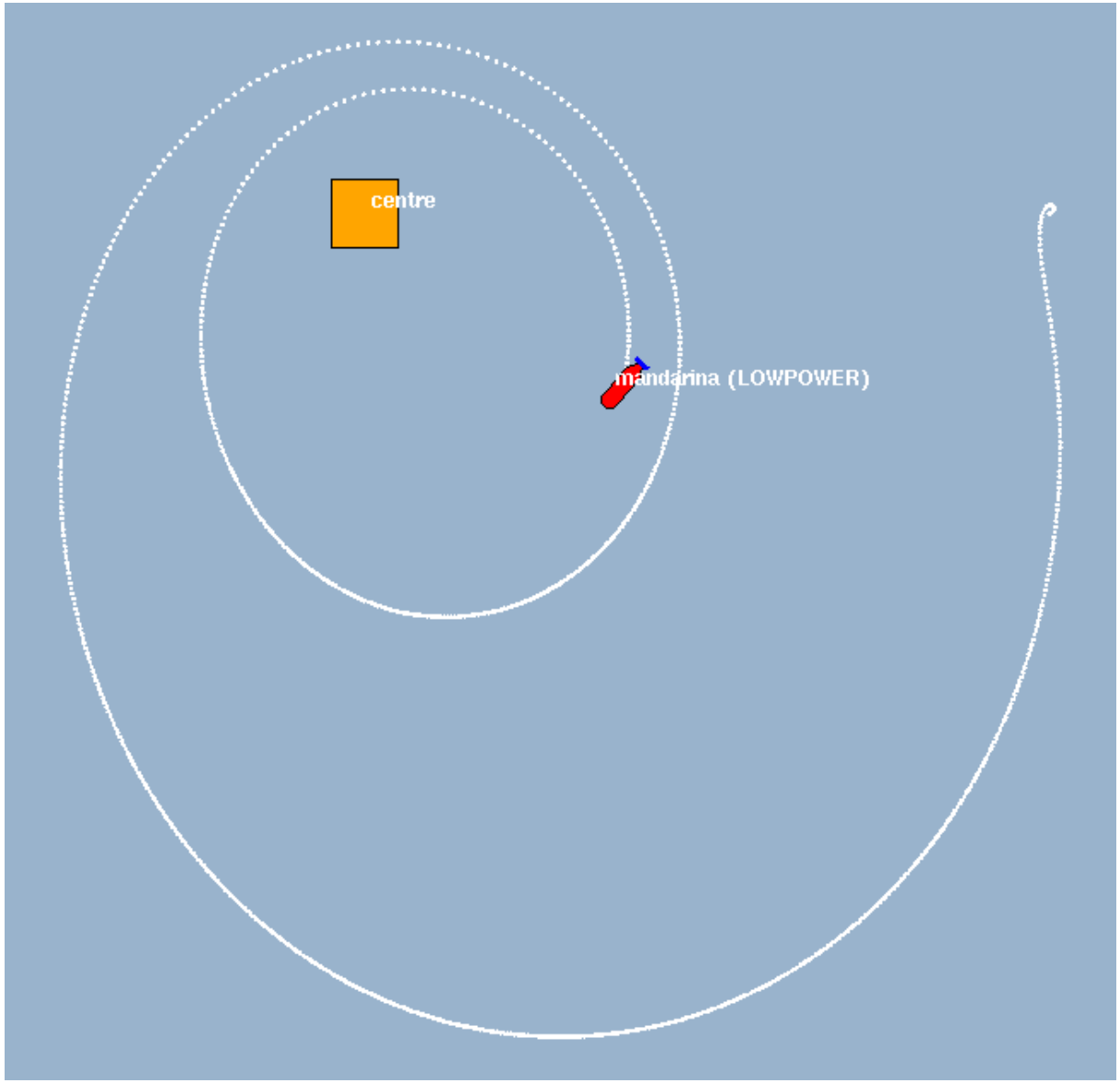
Example — behaviour

```
Behavior = BHV_ConstantHeading
{
  name = BHV_ConstantHeading_LowPower
  pwt = 100
  condition = (MODE == LOWPOWER)

  heading = 0
  updates = bhv-constant-heading-lowpower-updates
  duration = no-time-limit
}
```

Example — behaviour



Example — behaviour



Example — JANUS

- software defined acoustic modem
- FSK modulation / demodulation
- interleaving / de-interleaving
- convolutional encoder / Viterbi decoder
- runs on PC 104 stack on board OEX

pOctaver — when to use

- quick and dirty glue code
- behaviours
- rapid prototyping
- simulation

pOctaver — pros

- small, powerful, robust code snippets
- no compilation required
- no knowledge of C++ required
- testing directly in Octave
- increased productivity
- better workflow
- no license fees

pOctaver — cons

- yet another language ...
- more resource intensive than C++
- less portable
- not good on embedded systems

pAny

Question:

is it possible to design a similarly simple interface for 'any' C++ function?

Answer:

yes, it's called pAny, and I'm working on it.

Distribution

- pOctaver is a 'finished' product
- distributed to workshop participants
- please use it
- open to suggestions for improvement
- please give feedback!

Document Data Sheet

Security Classification		Project No.
		EPOW
Document Serial No.	**Date of Issue**	**Total Pages**
NURC-PR-2010-002	August 2010	33 pp.

Author(s)

Vermeij, A., Pastore, T.J.

Title

pOctaver, adding a scripting language to MOOS.

Abstract

NURC uses MOOS-IvP in a scientific research and rapid prototyping environment. Contributions to the open-source suite of MOOS-IvP software is readily made by programmers adept at C++ programming. However, the utility of MOOS-IvP's benefits can be expanded to additional users who normally develop their work in Matlab. Octave is an open source Matlab clone. We developed pOctaver, which can directly run a script written in Octave as a MOOS application. This has great advantages in allowing for rapid prototyping. A complex function or new concept developed and preliminarily tested in Matlab can be quickly inserted into an existing moos mission for system-level simulation. The subsequent step from simulation in the MOOS environment to running on the vehicle follows immediately. The cycle from developing a concept and testing it in Matlab to running it on a vehicle can be reduced to hours or minutes, as contrasted with a timeline normally measured in days or longer if a conversion from Matlab code to C++ were required. pOctaver is a path by which a researcher can test a new function or application without requiring a single line of C-code to be written. Of course there is always a place for well-written and computationally more efficient C++ programs, and Octave/Matlab can never compete with C++ if computation times are critical. The advantage of pOctaver from an organizational standpoint is that the efforts of the researchers and programmers can be decoupled and are no longer serially dependent prior to a first in-water test of a new concept. The design and implementation of pOctaver is discussed, along with some examples of how it has been used at NURC in recent months. This work was initially presented at MOOS-DAWG (MOOS Development and Applications Working Group) meeting, August 2010, Cambridge, Mass. USA This document consists of the abstract, the presentation slides from the working group meeting, and the pOctaver source files as an annex. The annex is distributed as a tar file which is separate from the current pdf document, but integral to this reprint.

Keywords

Autonomy, computer programming, software design

Issuing Organization

NURC
Viale San Bartolomeo 400, 19126 La Spezia, Italy

[From N. America:
NURC
(New York) APO AE 09613-5000]

Tel: +39 0187 527 361
Fax:+39 0187 527 700

E-mail: library@nurc.nato.int