A Guide to Artifact Searching using MOOSIvP

Andrew Shafer, Michael Benjamin
Dept of Electrical Engineering/Computer Science, MIT
Cambridge MA 02139
<ajshafer@mit.edu>, <mikerb@mit.edu>

September 6, 2007

***Abstract*— This is the abstract.**

I. Introduction

This document describes the use and development of the artifact search system developed as a Master's of Engineering thesis by Andrew Shafer at MIT. This document assumes that the reader has a familiarity with MOOS and IvP and understands how to use those tools (see [4], [1], [2], and [3]).

First, a bit of terminology. In this document, an "artifact" is an object of interest. An artifact can be any detectable, identifiable object. In a naval application this would commonly be some type of mine. In naval terminology, "mine-hunting" (or mine-sweeping) usually refers to the process of detecting mines and deactivating or destroying them. "Mine-searching," on the other hand, refers to simply mapping out the locations of detected mines for later deactivation/destruction. Therefore, this project is more properly an artifact searching system, rather than a mine-hunting system.

A "search area" is the geographic region that the user desires to search (see Figure I). This area is broken up into uniform, discrete cells that together constitute the "search grid" (see Figure I).

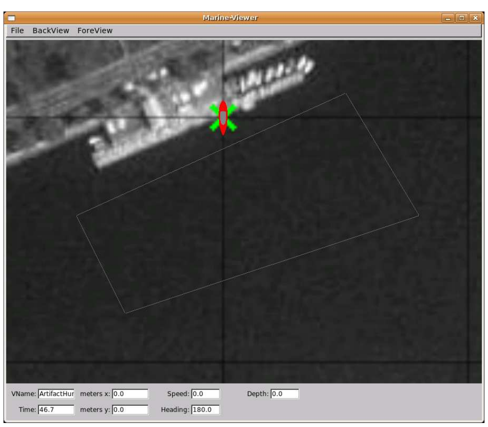

Fig. 1. A geographic area (a convex polygon) defined as a search grid.

There are two main MOOS processes and one IvPHelm behavior that implement the artifact search system. pSensorSim simulates the output of an imaginary sensor in a simulated artifact field. pArtifactMapper takes the output of pSensorSim, fuses it with output from other artifact search platforms in the area, and produces a likelihood map of artifacts in

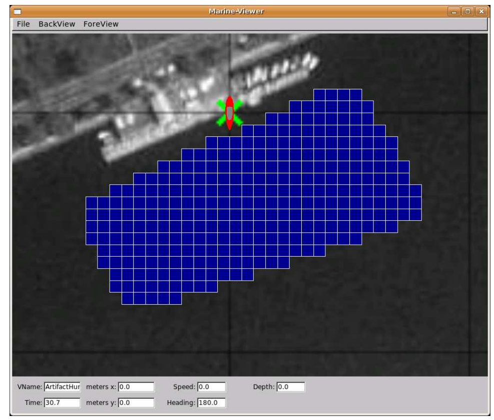

Fig. 2. A search grid defined over a search area.

the search region. The IvPHelm behavior, bhv_SearchGrid, provides desired heading and speed information to the helm to optimize the user's utility function (e.g. mapping an entire field with 95% confidence in the least amount of time).

II. pSensorSim

III. pArtifactMapper

IV. bhv_SearchGrid

V. Example Missions

References

[1] Michael R. Benjamin. *Interval Programming: A Multi-Objective Optimization Model for Autonomous Vehicle Control.* PhD thesis, Brown University, Providence, RI, May 2002.

[2] Michael R. Benjamin. Multi-Objective Navigation and Control Using Interval Programming. In *Proceedings of the Multi-Robot Systems Workshop*, NRL, Washington DC, March 2003.

[3] Michael R. Benjamin. The Interval Programming Model for Multi-Objective Decision Making. Technical Report AIM-2004-021, Computer Science and Artificial Intelligence Laboratory, MIT, Cambridge, MA, September 2004.

[4] Paul M. Newman. MOOS - A Mission Oriented Operating Suite. Technical Report OE2003-07, MIT Department of Ocean Engineering, 2003.

APPENDIX

Listing A.1 The MOOS File for Examples 1-3

Filename: alpha.moos

```
ServerHost = localhost
ServerPort = 9000
Simulator  = true
Community  = alpha
LatOrigin  = 42.3584
LongOrigin = -71.08745

//------------------------------------------
ProcessConfig = ANTLER
{
  MSBetweenLaunches = 200

  Run = MOOSDB          @ NewConsole = true
  Run = iMarineSim      @ NewConsole = true
  Run = pEchoVar        @ NewConsole = true
  Run = pLogger         @ NewConsole = true
  Run = pTransponderAIS @ NewConsole = true
  Run = pMarinePID      @ NewConsole = true
  Run = pMarineViewer   @ NewConsole = true
  Run = pHelmIvP        @ NewConsole = true
  Run = iRemote         @ NewConsole = true
}

//------------------------------------------
ProcessConfig = iMarineSim
{
  AppTick     = 4
  CommsTick     = 4
  MaxTransVel   = 3.0
  MaxRotVel     = 0.6
  StartLon      = 10
  StartLat      = -40
  StartSpeed    = 0
  StartHeading  = 180
}

//------------------------------------------
ProcessConfig = pHelmIvP
{
  AppTick    = 4
  CommsTick  = 4
  Domain     = course:0:359:360
  Domain     = speed:0:3:16

  Behaviors  = foobar.bhv
  VERBOSE    = terse
}

//------------------------------------------
ProcessConfig = pMarinePID
{
  AppTick    = 4
  CommsTick  = 4
  Verbose    = true

  DEPTH_CONTROL = false
  MAXRUDDER     = 100
  MAXTHRUST     = 100

  YAW_PID_KP               = 0.5
  YAW_PID_KD               = 0.0
  YAW_PID_KI               = 0.0
  YAW_PID_INTEGRAL_LIMIT   = 0.07

  SPEED_PID_KP             = 1.0
  SPEED_PID_KD             = 0.0
  SPEED_PID_KI             = 0.0
  SPEED_PID_INTEGRAL_LIMIT = 0.07
  SPEED_FACTOR             = 20
}

//------------------------------------------
ProcessConfig = iRemote
{
   CustomKey = 1 : HELM_VERBOSE @ "verbose"
   CustomKey = 2 : HELM_VERBOSE @ "terse"
   CustomKey = 3 : HELM_VERBOSE @ "quiet"
   CustomKey = 4 : DEPLOY       @ "true"
   CustomKey = 5 : RETURN       @ "true"
}

//------------------------------------------
ProcessConfig = pLogger
{
  AppTick       = 20.0
  CommsTick     = 20.0
  File          = alpha
  PATH          = ./datafiles
  SyncLog       = true @ 0.2
  AsyncLog      = true
  FileTimeStamp = true
  Log           = NAV_X     @ 0.1
  Log           = NAV_Y     @ 0.1
  Log           = NAV_Yaw   @ 0.1
  Log           = NAV_Speed @ 0.1
}

//------------------------------------------
ProcessConfig = pEchoVar
{
  AppTick   = 5
  CommsTick = 5
  Echo      = MARINESIM_X       ->  NAV_X
  Echo      = MARINESIM_Y       ->  NAV_Y
  Echo      = MARINESIM_YAW     ->  NAV_YAW
  Echo      = MARINESIM_HEADING ->  NAV_HEADING
  Echo      = MARINESIM_SPEED   ->  NAV_SPEED
}

//------------------------------------------
ProcessConfig = pTransponderAIS
{
  AppTick       = 2
  CommsTick     = 2
  VESSEL_TYPE   = KAYAK
}
```