uFldCollObDetect: Detecting Collisions with Obstacles

February 2021

Michael Benjamin, mikerb@mit.edu
Department of Mechanical Engineering
MIT, Cambridge MA 02139

1 Overview

The uFldCollObDetect application is run on the shoreside and maintains a current knowledge of all vehicle positions and obstacle locations and monitors for encounters between vehicles and obstacles that come within a certain range. The closest point of approach (CPA) is noted when the range between a vehicle to an obstacle transitions from closing to opening. Depending on the CPA value, one of three events may be declared, either an *encounter*, a *near miss*, or a *collision*, depending on user configured range parameters.

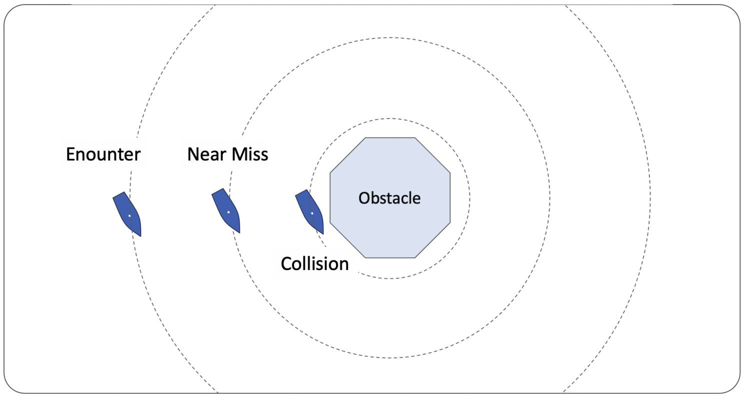

Figure 1: **Obstacle Collision Monitoring**: Three ranges are monitored. The *encounter* range is related to a range where there was likely a measure of obstacle avoidance in play by the vehicle. The *near miss* range indicates encounters that are noteworthy and should be very minimal over the course of a mission. The *collision* range is just above zero and constitutes an encounter that should warrant investigation of the mission.

2 Using uFldCollObDetect

2.1 Setting the Range Thresholds for Events

There are three range threshold parameters that may be set. The first, `encounter_range` is the CPA range beyond which a vehicle and an obstacle are considered to be too far away to be regarded as having had an encounter. Outside this range it's a non-event.

The second parameter, `near_miss_range`, determines a CPA range within which an encounter is considered to be a near miss. Encounters even closer, with the range specified by `collision_range`, are categorized as collisions.

The default values are:

- `encounter_range = 30`
- `near_miss_range = 10`
- `collision_range = 5`

The following relationship between parameters will be enforced:

`encounter_range` >= `near_miss_range` >= `collision_range`

If this ordering is not respected by the configuration parameters, then the following happens upon startup: (1) If the `near_miss_range` is smaller than the `collision_range`, the former is adjusted upward to the latter. (2) If the `encounter_range` is less than the `near_miss_range`, then the former is adjusted upward to the latter. (3) A configuration warning is generated and shown in the appcasting output.

Upon each encounter less than or equal to the `encounter_range`, an encounter counter is internally incremented. Likewise for near misses and collisions. A collision encounter will not however also increment the near miss counter. These counters are maintained globally for all vehicles and obstacles, as seen in lines 17-19 in the example appcasting output in Listing 3. They are also maintained *per vehicle*, as seen in lines 21-29 in Listing 3.

2.2 Setting Flags to be Posted Upon Events

Flags may be configured to be posted upon each event type - collision, near-miss or encounter. These flags are simply MOOS variable and value pairs like the flags in many other MOOS applications and helm behaviors. There are also macros available that can be filled in a run-time.
The following are the available macros for the *variable* portion of the flag:

- `$VNAME`: The name of vehicle involved in the encounter.
- `$UP_VNAME`: The uppercase name of the vehicle involved in the encounter.
- `$ID`: The ID or label of the obstacle involved in the encounter.

The following are the available macros for the *value* portion of the flag:

- `$VNAME`: The name of vehicle involved in the encounter.

- `$UP_VNAME`: The uppercase name of the vehicle involved in the encounter.
- `$ID`: The ID or label of the obstacle involved in the encounter.
- `$ENC_CNT`: The number of total encounters
- `$MISS_CNT`: The number of total near misses
- `$COLL_CNT`: The number of total collisions
- `$DIST`: The CPA range of the encounter
- `$CPA`: The CPA range of the encounter

If the `$DIST`, `$ENG_CNT`, `$COLL_CNT`, or `$MISS_CNT` macros are used in isolation in the message, the message will post as a double, not a string. This allows certain post-mission tools like `alogview` to generate time plots when posted as a double.

```
near_miss_flag  = NEAR_MISS_$UP_VNAME = $CPA
encounter_flag  = WARNING = Something happened between $VNAME and obstacle $ID
collision_flag  = CRITICAL = collision between $UP_VNAME and $ID
collision_flag  = $DIST
encounter_flag  = $ENC\_CNT
```

3 Configuration Parameters for uFldCollObDetect

The `uFldCollObDetect` application may be configured with a configuration block within a MOOS mission file, typically with a `.moos` file suffix. The following parameters are defined for `uFldCollObDetect`.

Listing 3.1: Configuration Parameters for uFldCollObDetect.

`collision_flag`: A MOOS variable and value posted upon the detection of a collision between a vehicle and an obstacle. Section 2.2.

`collision_dist`: The distance, in meters, within which the CPA between a vehicle and an obstacle is regarded as a collision. The default is 5 meters. Section 2.1.

`collision_range`: Same as `collision_dist`.

`encounter_flag`: A MOOS variable and value posted upon the detection of an encounter between a vehicle and an obsacle. Section 2.2.

`encounter_dist`: The distance, in meters, within which the CPA between a vehicle and obstacle is considered to be an encounter. The default is 30 meters. Section 2.1.

`encounter_range`: Same as `encounter_dist`.

`near_miss_flag`: A MOOS variable and value posted upon the detection of a near miss between a vehicle and an obstacle. Section 2.2.

`near_miss_dist`: The distance, in meters, within which the CPA between a vehicle and an obstacle is regarded as a near miss. The default is 10 meters. Section 2.1.

near_miss_range: The range, in meters, within which the CPA between a vehicle and an obstacle is regarded as a near miss. The default is 10 meters. Section 2.1.

near_miss_range: Same as near_miss_dist.

An Example MOOS Configuration Block

An example MOOS configuration block is provided in Listing 2 below. This can also be obtained from a terminal window with:

```
$ uFldCollObDetect --example or -e
```

Listing 3.2: Example configuration of the uFldCollObDetect application.

```
===============================================================
uFldCollObDetect Example MOOS Configuration
===============================================================

ProcessConfig = uFldCollObDetect
{
  AppTick   = 4
  CommsTick = 4

  encounter_flag = ENCOUNTER = $CPA
  collision_flag = COLLISION = $CPA
  near_miss_flag = NEAR_MISS = vname=$VNAME,obstacle=$ID,cpa=$CPA

  encounter_range = 30        // the default in meters
  near_miss_range = 10        // the default in meters
  collision_range = 5         // the default in meters
}
```

4 Publications and Subscriptions for uFldCollObDetect

The interface for uFldCollObDetect, in terms of publications and subscriptions, is described below. This same information may also be obtained from the terminal with:

```
$ uFldCollObDetect --interface or -i
```

4.1 Variables Published by uFldCollObDetect

The user may configure any number of postings (flags) to be generated upon an encounter, near miss or collision.

- APPCAST: Contains an appcast report identical to the terminal output. Reports are posted only in response to an appcast request messages.

4.2 Variables Subscribed for by uFldCollObDetect

The uFldCollisionDetect application will subscribe for the following variables:

- APPCAST_REQ: A request to generate and post a new apppcast report, with reporting criteria, and expiration.
- KNOWN_OBSTACLE: Information about a known obstacle.
- NODE_REPORT: A report on a vehicle location and status.

Obstacle information typically comes from uFldObstacleSim.

Command Line Usage of uFldCollObDetect

The uFldCollObDetect application is typically launched as a part of a batch of processes by pAntler, but may also be launched from the command line by the user. To see command-line options enter the following from the command-line:

```
$ uFldCollObDetect --help or -h
```

This will show the output shown below.

```
Usage: uFldCollObDetect file.moos [OPTIONS]

Options:
  --alias=<ProcessName>
      Launch uFldCollisionDetect with the given process
      name rather than uFldCollisionDetect.
  --example, -e
      Display example MOOS configuration block
  --help, -h
      Display this help message.
  --interface, -i
     Display MOOS publications and subscriptions.
  --version,-v
      Display the release version of uFldCollisionDetect.
```

5 Terminal and AppCast Output

The uFldCollObDetect application produces some useful information to the terminal and identical content through appcasting. An example is shown in Listing 3 below. On line 2, the name of the local community, typically the shoreside community, is listed on the left. On the right, "0/0(2426) indicates there are no configuration or run warnings, and the current iteration of uFldCollObDetect is 2426. Lines 4-9 show the how the application was configured at launch time.

Lines 11-15 convey the the overall application state, the total number of encounters, near missions and collisions (lines 12-14), and the closest that any vehicle has come to any obstacle ever (line 15).

Lines 17-25 show the current state of each vehicle with respect to each obstacle. These distances will go lower as the vehicle closes on an obstacle and then reset once the vehicle has cleared the obstacle. Lines 27-32 show the minimum distance, per vehicle, during the lifetime of the mission. Lines 35-43 show recent events, including reacent postings configured by the user through its flags.

Listing 5.3: Example terminal or appcast output for `uFldCollisionDetect`.

```
 1  ===================================================================
 2  uFldCollisionDetect shoreside                             0/0(2426)
 3  ===================================================================
 4  Configuration:
 5  ============================================
 6  Obstacles: 5
 7  Collision Dist: 1
 8  Near Miss Dist: 5
 9  Encounter Dist: 14
10  ============================================
11  State:
12  Total Encounters:  44
13  Total Near Misses: 14
14  Total Collisions:  0
15  Global Min Dist:   2.5
16  ============================================
17  Min Distances (for current encounter):
18  ============================================
19  obstacle  gilda  henry
20  --------  -----  -----
21  ob_0      14.7   14.6
22  ob_1      14.2   14.4
23  ob_2      14.9   14.9
24  ob_3      14.8   14.6
25  ob_4      14.6   14
26  ============================================
27  Min Distances (foe each vehicle, ever):
28  ============================================
29  Vehicle  Min Dist Ever
30  -------  -------------
31  gilda    2.5
32  henry    4.1
33
34  ===================================================================
35  Most Recent Events (8):
36  ===================================================================
37  [1046.54]: OB_TOTAL_ENCOUNTERS=44
38  [1046.54]: OB_TOTAL_NEAR_MISSES=14
39  [1046.54]: OB_ENCOUNTER=2.52
40  [1046.54]: OB_NEAR_MISS=2.52
41  [1046.54]: OB_NEAR_MISS_FULL=vname=gilda,ob=ob_0,dist=2.52
42  [1044.85]: OB_TOTAL_ENCOUNTERS=43
43  [1044.85]: OB_TOTAL_NEAR_MISSES=13
```