Mike McGinty
Bobby Darian

Ten words. Really. A Story of DCS World's RWR Display.

Problem: Simulator Limitations

DCS World doesn't support exporting some instrumentations to external monitors - the F-15's TEWS display in this case. We can export the data needed to recreate this particular display ourselves though, through DCS's Lua API.

The display itself is a 2D, top-down view centered on the player's aircraft, showing the relative positions (sorted by priority, based on likelihood of being a threat) of detected radar emissions. TEWS stands for "Tactical Electronic Warfare System", and RWR (Radar Warning Receiver) is the part of that system we're concerned with here - specifically the display.

Background: Time to youtube

First, the best introduction out there is a video here: http://youtu.be/xB6_MVFXyoY

Second, read the associated documents (start with the README) at https://github.com/mach327/DCS-RWR-racket

Analysis: Modeling a complex system is hard.

Draw the RIGHT pictures. In real time. From a simulator. Accurately per a fairly complex spec. That we don't necessarily have all the data for.

We'll use object orientation to describe the RWR system itself, and the various threats. Since the threats are organized in a taxonomy of types with four levels within the simulator itself, and the RWR system uses signatures of known radars to determine how to draw each signal (recognizing aircraft radars versus ground-based radars), an object oriented design lends itself well to the problem. We can have a 'generic' threat, airborne, sea, and land threats can inherit from the generic, and so on, each with their own traits as needed to simulate a realistic TEWS display.

Data set: As delivered over a TCP connection to or from the simulator.

In this case, we're working with 'live' data exported from the simulator, in JSON format, describing the RWR system and received radar signals. A status update is sent per 100ms of simulated time.

We can also save and replay this data for testing purposes, using common linux tools (netcat, pipeviewer).

Deliverables: A working, accurate display.

For the demonstration, a live simulator view would be preferable. If it's running the display off the Raspberry Pi and a little 4"x3" lcd screen Mike has, all the better.

If a live simulator is not possible (space, time constraints, bad weather, etc), then a video of the simulator view and expected TEWS display output can be rendered and saved, and the captured data can then be replayed in synch with the video.

Accuracy of the display can be determined with comparisons to the in-game rendered display, and with comparisons to reference data (preferably from the real TEWS system documentation, but this is unlikely). For reference data, it may also be helpful to look up A-10 RWR documents as the systems are somewhat alike in how they display radar threats.

Evaluation: A reliable, working, and accurate display is the goal.

Evaluation of the end product should be focused on accuracy of the systems simulated by DCS World, or as reported in reference documents, with a preference towards reference documents.

A display like this should feel easy and useful, without being a pain.

Schedule:

	What Mike Will Do	**What Bobby Will Do**
First: 11/24	Gather more resources for RWR symbology and behavior, structure code for good organization. Figure out why tcp-read is blocking when there's plenty to read. Classify radar types. Lookup table for proper symbology (e.g. F-15C -> 15) Fix coordinate differences between DCS and racket.	

Second: 12/1	Make sure documentation is up to snuff. Bugfixes. Make sure it works on the Raspberry Pi. All the X windowing system stuff required for the RPi.	
Final: 12/8	Final project. Should look similar enough to DCS RWR or real RWR display to be usable and realistic - e.g. read the docs, and be able to use our RWR display.	