Federal Aviation
Administration

DOT/FAA/AM-19/11
Office of Aerospace Medicine
Washington, DC 20591

Exploring General Aviation Pilots' Response to Graphical Presentation of Probabilistic Weather

William R. Knecht

FAA Civil Aerospace Medical Institute
Oklahoma City, OK 73125

Final Report

Technical Report Documentation Page

1. Report No.	2. Government Accession No.	3. Recipient's Catalog No.
DOT/FAA/AM-19/11		

4. Title and Subtitle	5. Report Date / 6. Performing Organization Code
Exploring general aviation pilots' response to graphical presentation of probabilistic weather	5. Report Date: September 2019 6. Performing Organization Code:
7. Author(s): Knecht W	8. Performing Organization Report No.
9. Performing Organization Name and Address: FAA Civil Aerospace Medical Institute P.O. Box 25082 Oklahoma City, OK 73125	10. Work Unit No. (TRAIS) 11. Contract or Grant No.
12. Sponsoring Agency name and Address: Office of Aerospace Medicine Federal Aviation Administration 800 Independence Ave., S.W. Washington, DC 20591	13. Type of Report and Period Covered 14. Sponsoring Agency Code

15. Supplemental Notes

16. Abstract

Probabilistic weather displays depict the chance of encountering heavy weather at a given place at a given future time. As such, they offer more information than deterministic displays, which merely show the results of running one single weather model one single time. However, the issues involved in the use of these displays are complex, and it would be a mistake to assume that pilots can quickly learn to interpret and use them effectively. Unlike most meteorologists touting such displays, most pilots do not have formal training in probability theory or application. Moreover, it is human nature for most of us to believe that we understand complex phenomena better than we actually do.

To put such a device to the test, we created a part-task computer simulation of a looping probabilistic weather forecasting display, a looping deterministic forecasting display, and a historical-weather-only display (similar to current looping NEXRAD, showing only past weather). We then tested 18 general aviation pilots using the three displays on nine weather scenarios.

As a group, pilots reported that they felt able to navigate around heavy weather with 10-minute weather updates and 30 minutes' lookahead on the two forecasting displays, but would have preferred 5-minute updates. They preferred the two forecasting displays significantly more than the looping-NEXRAD-type display, and felt the forecast displays would keep them significantly safer, with the probabilistic display superior to the deterministic, although not significantly so. They felt that all three display types took about the same amount of mental effort to use, and would require about the same amount of training and practice to achieve proficiency.

As individuals, analysis of individual differences, however, showed considerable variability in these preferences and opinions. In other words, not every pilot preferred the forecasting displays. Nor did all think they would be effortless to use and quick to master.

Finally, a logical and mathematical analysis of the factors involved in displaying and using probabilistic weather displays reveals a substantial number of non-trivial challenges. The issue of whether or not these can be successfully mitigated by technology and training will require further investigation. It does seem possible that inclusion of the ability to "scroll back and forth, forward in time," plus addition of a range ring around the aircraft icon may be sufficient, and sufficiently easy to learn, for the average pilot to safely avoid heavy weather.

17. Key Words	18. Distribution Statement
aviation weather, probabilistic weather, mathematically generated NEXRAD, weather displays	Document is available to the public through the Internet: http://www.faa.gov/go/oamtechreports/

19. Security Classif. (of this report)	20. Security Classif. (of this page)	21. No. of Pages	22. Price
Unclassified	Unclassified	56	

Form DOT F 1700.7 (8-72) Reproduction of completed page authorized

Exploring general aviation pilots' response to graphical presentation of probabilistic weather

William R. Knecht
Civil Aerospace Medical Institute, FAA, AAM-510

EXECUTIVE SUMMARY

Probabilistic weather displays depict the chance of encountering heavy weather at a given place at a given future time. As such, they offer more information than deterministic displays, which merely show the results of running one single weather model one single time. However, the issues involved in the use of these displays are complex, and it would be a mistake to assume that pilots can quickly learn to interpret and use them effectively. Unlike most meteorologists touting such displays, most pilots do not have formal training in probability theory or application. Moreover, it is human nature for most of us to believe that we understand complex phenomena better than we actually do.

To put such a device to the test, we created a part-task computer simulation of a looping probabilistic weather forecasting display, a looping deterministic forecasting display, and a historical-weather-only display (similar to current looping NEXRAD, showing only past weather). We then tested 18 general aviation pilots using the three displays on nine weather scenarios.

As a group, pilots reported that they felt able to navigate around heavy weather with 10-minute weather updates and 30 minutes' lookahead on the two forecasting displays, but would have preferred 5-minute updates. They preferred the two forecasting displays significantly more than the looping-NEXRAD-type display, and felt the forecast displays would keep them significantly safer, with the probabilistic display superior to the deterministic, although not significantly so. They felt that all three display types took about the same amount of mental effort to use, and would require about the same amount of training and practice to achieve proficiency.

As individuals, analysis of individual differences, however, showed considerable variability in these preferences and opinions. In other words, not every pilot preferred the forecasting displays. Nor did all think they would be effortless to use and quick to master.

Finally, a logical and mathematical analysis of the factors involved in displaying and using probabilistic weather displays reveals a substantial number of non-trivial challenges. The issue of whether or not these can be successfully mitigated by technology and training will require further investigation. It does seem possible that inclusion of the ability to "scroll back and forth, forward in time," plus addition of a range ring around the aircraft icon may be sufficient, and sufficiently easy to learn, for the average pilot to safely avoid heavy weather.

INTRODUCTION

Research Mandate

The FAA NextGen Program is researching probabilistic weather information in the cockpit, with the goal of providing the best possible weather information to pilots aloft (Guinn & Barry, 2012). Probabilistic representations have many human factors implications. For one, we lack knowledge about whether providing information about forecast uncertainty can or will reliably improve weather-avoidance decision making by pilots. There is fairly strong indication that the general public do not clearly understand probabilistic-weather basics such as *probability of precipitation* (PoP[1], Morss, Demuth, & Lazo 2008). Probabilistic weather information is also the product of complex model-based algorithms, and it is unclear the extent to which pilots can or will trust and utilize this information. The trust problem arises from both its probabilism (namely, that the success/failure of any one single decision does not validate probabilistic information)

[1] PoP is the familiar "percent chance of rain," given by forecasters. This is a) **defined strictly** as PoP = C ×A, where "C" is the *confidence* that precipitation will occur somewhere in the forecast area during the time period specified, and "A" is the *percent of that area* that will receive measurable precipitation (≥ 0.01"), also b) **defined loosely** as the "point forecast", or percentage likelihood of ≥ 0.01" precipitation *at any one particular point* in the forecast area during the time period specified (NWS, 2018). These two definitions are actually mathematically equivalent.

and the hard-to-understand automation required to generate it. Additionally, at the human information processing level, basic research has shown that humans have problems in utilizing probabilistic information.

This project supports the Weather Technology in the Cockpit Program (WTIC Program) objective of developing Minimum Weather Services (MinWxSvc) for Part 121/135 and Part 91 that define the minimum weather information needed in the cockpit and its associated parameters, minimum rendering standards for the information, and enhanced weather training. Psychophysical and cognitive investigation of probabilistic weather presentations can identify stimulus information necessary and sufficient to enable and increase safe, efficient convective weather avoidance. Human factors research can then test theory-guided weather presentations to verify safe and efficient use by pilots. Knowing these results, the FAA can offer sound, proactive, human factors-guided product design advice to manufacturers. We will also learn what, if any, product design features can benefit from formal regulation/certification. Finally, we can design training products to proactively counter risks that have been identified, but which fall outside the domain of regulation.

Background

Current State of the Art in Weather Information Display

Historical-information display. Currently, the most-advanced convective weather information display in common flightdeck usage is the looping NEXRAD format. This shows an hour or two's *historical* radar-reflectivity images, sampled about every five minutes, superimposed over a map of terrain and/or roads, and then displayed sequentially at a frame rate intended to impart a sense of *apparent motion* to both storm and aircraft icon. The pilot's job is then to mentally extrapolate the future position of both, and use those imagined positions to navigate safely around or through the storm system.

Of course, mental extrapolation has no guarantee of accuracy. Storms rarely develop exactly as we imagine. Adding to the challenge is the problem of *latency*, or time delay in delivering the radar information. It takes about 5-10 minutes for the NWS to assemble, process, and distribute the basic NEXRAD frames. Compounding this problem, most pilots use secondary information providers who add their own enhancements to the NWS feed, further adding to the time delay. Ultimately, the total typical information latency is generally around 10-15 minutes, ensuring that even the most-current radar images are far from real-time.

The FAA's response. A primary purpose of the Federal Aviation Administration's WTIC Program is to support development of MinWxSvc standards for the display of weather information to pilots aloft. The WTIC Program seeks to advance the state of the art in weather-information display by investigating the human factors involved, helping to open the way to solutions for any difficulties found associated with any given technology.

Forecast-information display. Prior research shows that historical weather information such as NEXRAD, while helpful, needs to be supplemented by predictive, *forecast*, information (Knecht, 2016). Modern computerized weather models offer substantial accuracy in the 30-60-minute lookahead range critical to tactical weather avoidance[2]. This makes predictive modeling a leading candidate for advancing the state of the art in weather technology in the cockpit.

Advancing the State of the Art

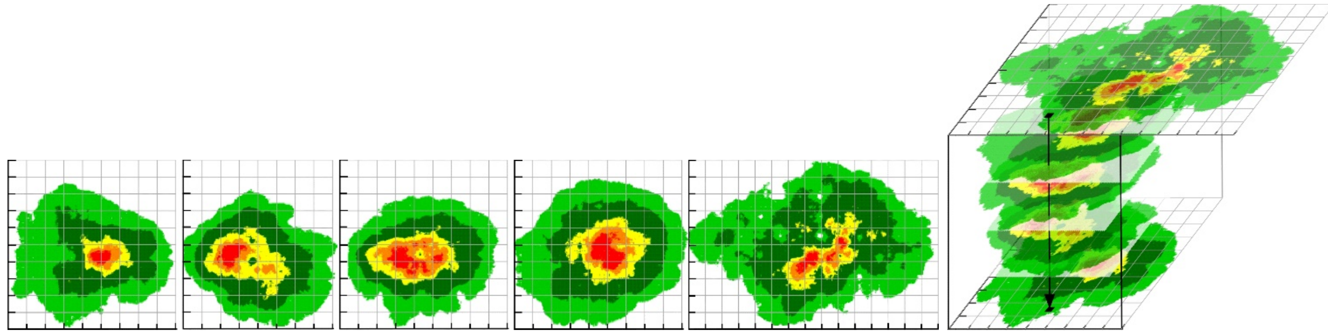
Deterministic weather forecasting. Deterministic forecasting essentially involves running one, single computer weather model one, single time, and then displaying the results as a NEXRAD-like image representing decibels (dBZ) of radar energy reflected by rain and hail.

Obviously, the single-model/single-run approach makes the deterministic forecast relatively fast-running, but at a cost of informational content, since different weather models give somewhat different results, even given identical input.

Probabilistic weather forecasting. In contrast to deterministic forecasting, probabilistic forecasting involves the generation of many forecasts, either by slightly varying the input parameters of a single model each time it is run, or by running many different models. Either way, we can then "stack" the multiple outputs for a given region and time, as if overlaying them to see what models agree in a given small location on the map. Figure 1 illustrates. The percentage of models that predict heavy convective weather (>40 dBZ) in that small location then becomes a concurrence metric. The higher the

[2] Personal conversations with the Associate Director of the Center for Analysis and Prediction of Storms (Brewster, 2016) concerning modeling accuracy led us to conclude that 1 hour forecast lookahead could be both sufficiently accurate and provide sufficient time for pilots to avoid most heavy weather.

percentage agreement (concurrence), the more likely we can consider *that location* to end up experiencing heavy weather at *that time*.



a b

Figure 1. Probabilistic weather modeling. a) Five different weather models give five different predictions for storm intensity; b) "Stacking" the results shows where the models agree and disagree, producing a percentage-agreement grid representing *the probability of encountering heavy weather in that place at that time*. Note that, because the final probabilistic depiction is a *sum of positive values*, its area will always be bigger than the area of any of its separate parts.

Because many model runs are conducted, probabilistic weather forecasting will clearly run slower than deterministic forecasting. However, the probabilistic approach generates crucial information about the *likelihood* of encountering heavy weather. And, given today's supercomputers, the technology does appear sufficiently fast to deliver timely forecasts in the 30-60-minute range.

Human factors concerns. One unassailable principle we learn from human psychology is that people typically do not behave like perfect information-processors. In fact, people often overestimate their own abilities (Kruger & Dunning, 1999; Ames & Kammrath, 2004), exhibit biases in their decisions, and use simple rules (*heuristics*) rather than more complicated calculation of probabilities, costs, and benefits (Kahneman, Slovic, & Tversky, 1982). Applying this principle to probabilistic weather forecasting, we would be making a mistake to assume that most pilots will find probabilistic displays as useful and easy to understand as meteorologists would like them to. The human factors of this are an empirical issue, and must be tested.

This testing is going to involve such questions as:

1. Historical vs. forecast	Is display of historical information necessary and sufficient to safely avoid heavy weather? What about forecast information? Would displaying both be best?
2. Update frequency	How often should weather images update (e.g., every 5 minutes? 10 minutes?)
3. Lookahead	How far would pilots need to see into the future to reliably avoid heavy weather?
4. Pilot acceptance	How enthusiastically do pilots accept a new way of performing an old task?

Clearly, no one study can address all these questions. A series will be necessary. We therefore begin by describing the methodology of the present study, a beta-test of a part-task simulation designed to quickly and inexpensively investigate key human-factors issues associated with the display of probabilistic weather information to pilots aloft.

METHODS

General Approach

Investigating effects of adding forecast information to existing NEXRAD

Two high-level dimensions to MinWxSvc for cockpit weather information are *what weather information should be shown to pilots* and *how often should it be updated?* Specifically, this means things like what information do historical radar images contain, what information do forecast images contain, and what should our minimum temporal image update rate be?

What should be shown? This was our first key human factors concern. Essentially, we wanted to test three weather information types:

1. Historical — Standard looping NEXRAD, similar to what exists today
2. Deterministic forecast — A forecast based on running a single weather model (explained below)
3. Probabilistic forecast — A forecast based on running an ensemble of weather models (explained below)

It made sense to use current looping NEXRAD as the base, or *control condition*, against which forecast weather could be compared. We could then add the additional information to that base, resulting in three test conditions, or *Weather Information* combinations[3].

1. Historical-only (**H-only**)
2. Historical + Deterministic forecast (**H+D**)
3. Historical + Probabilistic forecast (**H+P**)

In the H+D condition, the same color scheme could be used for historical and forecast weather, since both represented dBZ reflectivity. In the H+P condition, sponsors suggested we represent the forecast frames using the Corridor Integrated Weather System (CIWS) scheme developed by M.I.T. Lincoln Laboratories (Evans, Carusone, Wolfson, Crowe, Meyer, and Klingle-Wilson, 2002). Figure 1, row 3 below will illustrate.

How often should forecast frames be updated? This was a second key human factors concern. Along with the *number of frames* displayed, *frame intervals*—the length of time between successive frames—will determine size of the temporal window, namely how far pilots can "see into the past and future," also known as *lookbehind* and *lookahead*.

Given the large number of possible combinations for both number of frames and frame interval, an experiment could quickly grow unwieldy. It therefore made sense to be minimalistic, and select a floor (a minimum number) for both the number of frames and the frame interval, and then test whether that combination resulted in adequate pilot decision making.

Specifically, look*behind* time and number of frames should be sufficient for the user to recognize *trends* in weather direction and speed. Look*ahead* time and frames should be sufficient such that a safe maneuver around weather could be conceived—keeping in mind that Advisory Circular AC 00-24c (FAA, 2013) suggests avoidance of heavy weather (>40 dBZ radar return) by at least 20 nm—yet stay within the 1-hour time window where forecasts are likely to remain sufficiently accurate to support tactical navigation.

So, first, to investigate frame interval, we chose updates 10 simulated minutes apart, rather than the usual 5 minutes, to see if pilots could still accomplish the weather-avoidance task.

The next practical question was how many historical frames to display? Showing too few would make the sense of apparent motion too weak. Showing too many would make the entire frame loops ponderously long.

Knowing that apparent motion can be induced with as few as two frames (Mather & Challinor, 2009) allowed us to compromise and limit the number of historical frames to just three. This meant that Historical-only scenarios contained just three images, as the top row of Figure 2 shows.

Choosing the number of forecast frames for Historical + Forecast scenarios was now fairly easy. The middle and bottom rows of Figure 2 illustrate. If pilots could see 30 minutes "backward in time," then they should also be able to see at least 30 minutes "forward in time" as well. This meant at least three forecast frames should be shown—plus a fourth frame to represent "time = 'now,'" which would technically be a forecast because of the latency factor described earlier. This "4×10" time span fell within the 60-minute temporal window the National Oceanic and Atmospheric Agency (NOAA) currently calls "nowcasting." And, it would allow about 45-60 miles' travel at typical GA speeds (depending on winds aloft)—a reasonable distance to consider in small-aircraft weather-avoidance.

[3] Technically, this is called the *subtraction method*, because it gives us the opportunity to statistically subtract the sampled effects of Historical information from both the Historical + Deterministic and Historical + Probabilistic conditions (e.g., H+D-H=D). There is, of course, an interaction effect, so the full representation would be H+D+H×D-H=D+H×D.

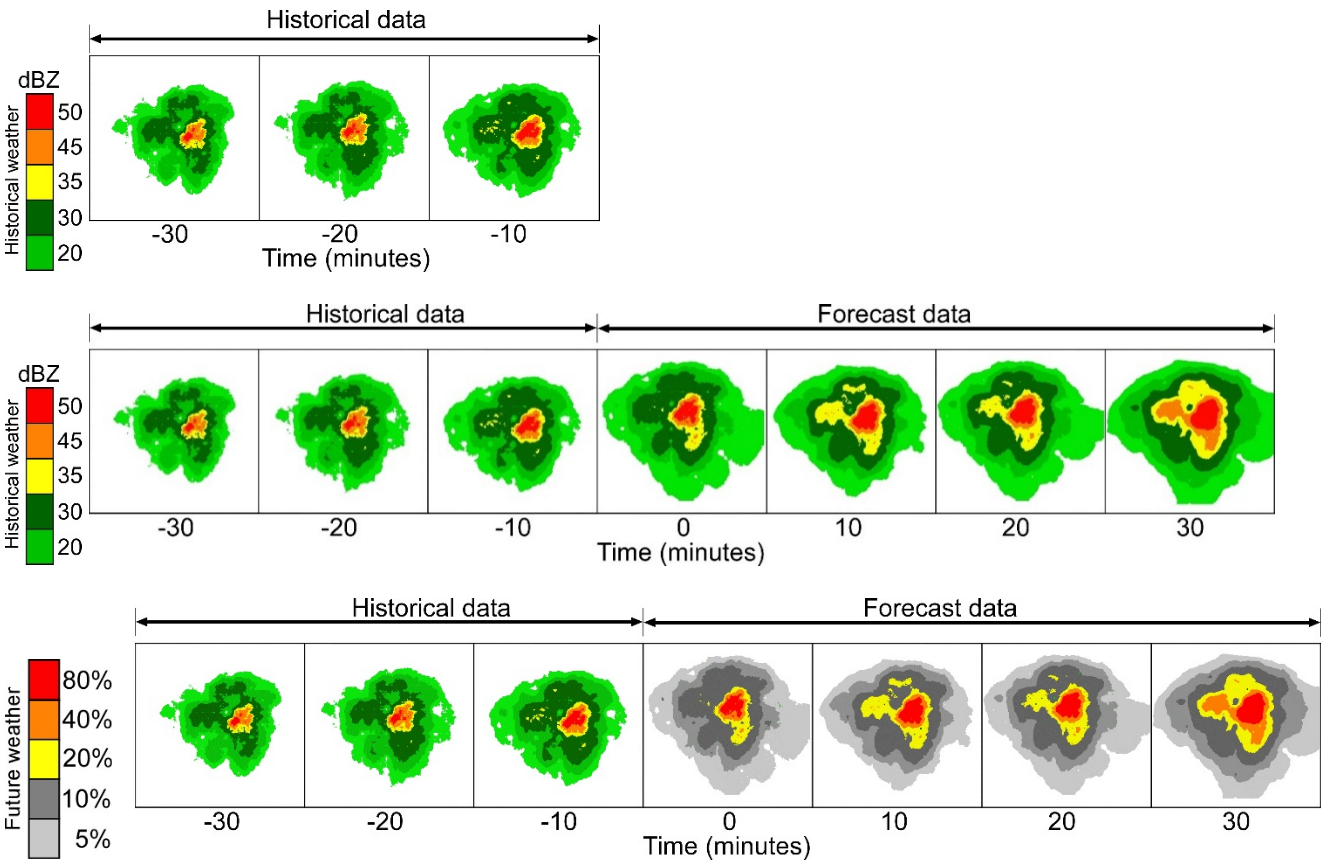


Figure 2. Top row) The color scale and three historical-weather frames of an H-only scenario, Middle row), the three historical-weather frames plus four deterministic-forecast frames of an H+D scenario, Bottom row) the three historical frames plus four probabilistic-forecast frames of an H+P scenario.

Evolving storm size. A final basic question of interest concerned what pilots react to when they see a set of looping weather images. Do they respond only to the storm's interior heavy weather (>40 dBZ reflectivity)? Or, do they also instinctively respond to the overall exterior shape of the storm, trying to avoid the entire outermost boundary altogether? This issue had gained importance due to the results of a previous WTIC Program-sponsored study (ATSC, 2013), which suggested that pilots may respond more vigorously to probabilistic weather depictions merely because their final size will logically always be larger than that of any of the individual deterministic parts summed to create the probabilistic whole.

That that end, we set up three ways in which storm cells would evolve in exterior size over time:

1. Size increasing over time
2. Size remaining constant
3. Size decreasing over time

Figure 1, in fact, shows a storm increasing in size over time.

Independent Variables. The above discussion is technically a description of two independent variables (IV)—*Weather Information* combination (with three test conditions) and *Storm Growth* type (also with three test conditions). Together, these lead to a total of nine possible weather-scenario combinations:

1. Historical-only scenarios
    a. Storm cell size increasing over time
    b. Storm cell size static over time
    c. Storm cell size decreasing over time
2. Historical + Deterministic-forecast scenarios
    a. Storm cell size increasing over time
    b. Storm cell size static over time
    c. Storm cell size decreasing over time
3. Historical + Probabilistic-forecast scenarios
    a. Storm cell size increasing over time
    b. Storm cell size static over time
    c. Storm cell size decreasing over time

These nine combinations would be tested using a *repeated-measures design*, meaning that each pilot in the experiment would see all nine conditions.

Hardware and Software

Hardware

Hardware consisted of a Dell Latitude E6440 laptop computer with optical mouse.

Software

Technical computing system. All code was written in *Mathematica 11.3* (Wolfram, 2018), a technical computing system encompassing a broad range of mathematical computation, image processing, graphic user interface (GUI) development, and file-handling capabilities, making it a viable platform for relatively simple part-task simulation such as ours.

Two main programs were written, the Weather Generator, and the Part-Task Simulator.

The Weather Generator. The Weather Generator was complex. Its detail is therefore moved back to Appendices A (method) and B (code) to simplify and highlight this report's more-important findings.

To briefly summarize, the Weather Generator was written to create sets of relatively simple NEXRAD-looking weather radar stimuli, for instance the images in Figure 1. This was accomplished by first generating *keyframes*—the first and last images of a series—and then smoothly morphing between them to create sets of images that looked as if they evolved continuously from one to the next.

These keyframes were generated by a variant of the *diamond-square algorithm*, a fractal, recursive computer function first used extensively to create artificial terrain in computer games. "Fractal," here means that the function started out operating on an area the size of the entire keyframe, then divided that into quarters, then each quarter into sixteenths, and so on, repeatedly, until it got down to the level of individual pixels. "Recursive" means a function that calls itself repeatedly, in this case until a 513×513 array in memory was filled with numbers representing "heights," constituting a *heightmap*, which could then be assigned colors by a *color function*.

The heightmap and color function were integral to controlling both the total number of pixels assigned to each keyframe, *and* their relative ratio of "heavy weather pixels" (orange and red). In this manner, we could create "weather" that could change overall size while keeping the size of its interior heavy weather constant to within 1% in both keyframes.

Each scenario actually required generating four sets of weather images. The reason for this will be described later. For now, note that historical-only scenarios had 3*4 = 12 images in total, while deterministic and probabilistic scenarios had 7*4 = 28.

The Part-Task Simulator (PTS). The purpose of the PTS was to play back image series created by the Weather Generator, and to record pilot reactions such as the initiation time and nature of maneuvers. *Mathematica* code is presented in Appendix C.

Figure 3 shows an annotated screenshot of a probabilistic weather scenario. Historical-only and deterministic scenarios looked similar, but had a background showing "H" or "D," and showed color scales appropriate to the weather being displayed.

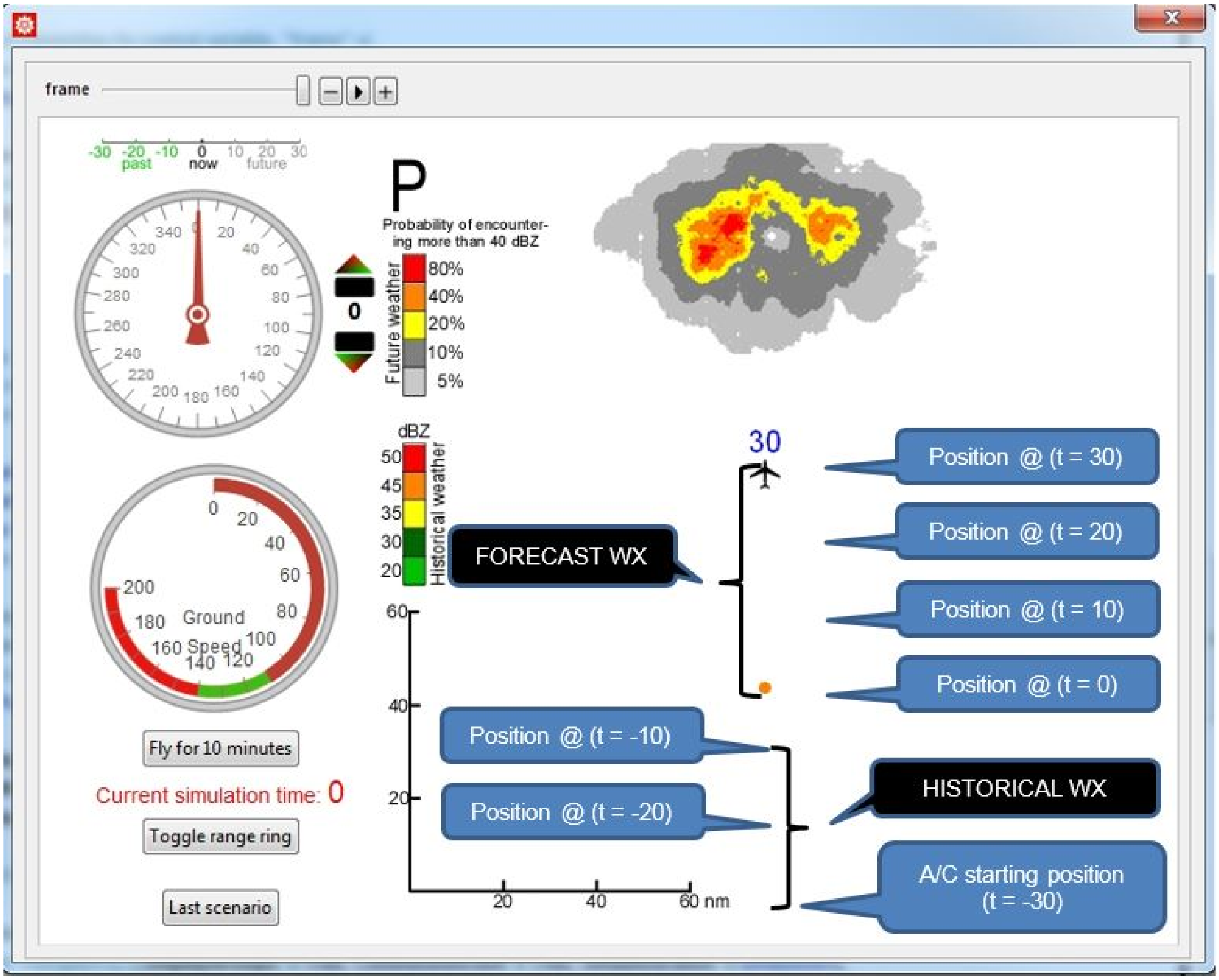


Figure 3. Full-size, annotated screenshot of the PTS, showing probabilistic weather information.

During each scenario the aircraft icon always started near the bottom of the PTS screen, heading straight up at 360/0° (aeronautical coordinates), while the weather always started directly above the aircraft, at the top of the screen, moving straight down at 180°. This emulated a track-up display, giving the impression that the aircraft was heading directly toward the weather at the start of each scenario.

"Heading" was controlled by mouse-clicking on the control set just to the right of the heading gauge. One click on the top/bottom black rectangle increased/decreased heading by 5°; one click on an adjacent triangle increased/decreased heading by 1°.

"Speed" was controlled by directly clicking on the bottom gauge at the point representing desired speed. "Speed" was denoted as ground speed, merely to simplify the task and keep focused on the method of weather display. We were mindful of the fact that actual ground speed is typically displayed in only the most sophisticated aircraft, and so mentioned this to pilots before starting data collection (none objected).

A "Toggle range ring" button allowed turning a 20-nm scale-radius range ring on and off. Previous research has shown that a range ring is theoretically necessary for maintaining accurate separation from heavy weather (Knecht, 2016), and the current experiment provided an opportunity to check pilot preferences for its use.

A slider labeled "frame" sat at the very top-left of the PTS. This allowed pilots to literally "scroll back and forth in time," emulating a feature first conceived by NASA personnel (Wu, Luna, & Johnson, 2013). Our frame-slider had a Pause/Advance toggle button to its left, which stopped/started frame looping. The Pause/Advance itself was bracketed by a go-back-one-frame button (-) and a go-forward-one-frame button (+), to allow precise “control over time,” if desired.

When finished with each scenario, a button at the bottom-left allowed pilots to advance to the next. Upon reaching the ninth scenario, the button displayed “Last scenario” to notify them they were nearly finished with the experiment.

Finally, the last control was a “Fly for 10 minutes” button just under the speed gauge. This allowed pilots to “jump forward 10 simulated minutes,” and will be described in greater detail momentarily.

Pilot’s task. The pilot’s central task was to maneuver left or right to continually maintain at least 20 scale nm from the ever-changing, ever-approaching heavy weather.

Each scenario began in looping mode, with actual frame rate set at 0.7 frames/sec, and depicting 10-minute intervals between frames. All scenarios began by showing three successive frames of historical weather (30 minutes lookbehind). Historical scenarios consisted of only those three frames. In the deterministic and probabilistic scenarios, the three historical frames were followed by four forecast frames (“now,” plus 10, 20, and 30 minutes lookahead). With each new frame, the aircraft icon moved to a new position appropriate to its current heading and speed. Right above the aircraft icon, in blue numerals, we displayed the simulated elapsed-time-since-scenario-start of that particular frame.

The “Fly for 10 minutes” button. Figure 3 shows this button right under the speed gauge. Pressing this button advanced the simulated elapsed time 10 minutes, and moved the weather and aircraft icon to their new, appropriate positions.

This capability emulated method previously used by NASA researchers Wu, Luna, and Johnson (2013) and shown to be practical for two reasons. The first was to take into consideration how pilots would actually use a weather display such as this. In real life, most would see the latest weather-forecast information, consider the situation, make a maneuver, and then fly a constant course, waiting for the next update.

The button’s second purpose was to speed up data collection, thereby helping to maintain pilot focus and avoid the fatigue and boredom pilots would have experienced, had they actually been forced to endure nine separate 30-minute scenarios. Without this capability, a repeated-measures experimental design with nine long scenarios in a single sitting would be difficult to justify.

During each scenario, pilots were given three chances to consider the situation, maneuver, and then jump ahead 10 minutes. Given the setup, this effectively took them just close enough to the storm cell to be able to calculate point-of-closest-approach (PCA), although derivation of PCA was not specified in the sponsor’s task list.

The ability to maneuver at t = 0, 10, and 20 minutes required four separate sets of images. Table 1 illustrates.

Table 1. What weather images were shown at each time period.

Time	Historical-only	Historical + Forecast
0	t = -30, -20, -10	t = -30, -20, -10, 0, 10, 20, 30
10	t = -20, -10, 0	t = -20, -10, 0, 10, 20, 30, 40
20	t = -10, 0, 10	t = -10, 0, 10, 20, 30, 40, 50
30	t = 0, 10, 20	t = 0, 10, 20, 30, 40, 50, 60
Total images per scenario	**12**	**28**

The reason we could not simply generate one set of images for t = -30, -20, -10, 0, 10. 20. 30, 40, 50, 60, and then just display them for the forecast scenarios was because, in a real situation, every 10 minutes pilots would see a *new, updated set of forecasts*, each new set looking slightly different from the previous one because of prediction error. Therefore, every updated set of ours also had to look slightly different from its predecessor. Appendix A shows specifically how this was done.

Experimental Design

This section details the independent and dependent variables, and describes control for unwanted experimental effects such as fatigue and learning. Sponsors requested we emphasize qualitative analysis. Therefore, dependent variables will be supplemented by *themes* and *anecdotal observations*, explained in more detail below.

Independent Variables (IV)

The IVs. Independent variables are factors we manipulate in an experiment to see how they affect designated outcomes (the dependent variables). The current design involved two IVs in a 3 × 3 = 9 total-trial design:

A. *3 Weather Information Types*:

1.	Historical	**H**	(Fig. 1, top row)
2.	Historical + Deterministic	**H+D**	(Fig. 1, middle row)
3.	Historical + Probabilistic	**H+P**	(Fig. 1, bottom row)

B. *3 Storm Size-Evolution Types:*

1.	Getting smaller over time	**SM**
2.	Static	**ST**
3.	Getting larger over time	**LA**

Repeated measures. A *within-participants* (repeated measures) statistical design was used. Each pilot therefore saw and responded to all nine weather information scenarios. This type of design is statistically powerful because each participant "serves as his or her own control," allowing individual error effects such as personality traits to be treated as constant across trials, allowing them to be mathematically subtracted during statistical analysis.

Control for time-related effects. Nonetheless, repeated-measures designs can suffer from time-related effects such as learning and/or fatigue effects. Consequently, participant performance may systematically improve or decline over time.

To control for such unwanted effects, we typically either counterbalance the presentation order of treatments, or randomize it. For example, to satisfy a counterbalanced design with 3 treatments (e.g., **H, D, P**) takes 3! (3 factorial = 3×2×1) = 6 participants if each participant gets one set (**HDP, HPD, DHP, DPH, PHD, PDH**).

In the current experiment, it was impossible to test enough participants to satisfy a fully counterbalanced design (9! = 362,880). Moreover, seeing all nine treatment combinations in purely random order might well be very confusing to pilots. Consequently, we settled on a hybrid design, with *Information Type* counterbalanced while *Storm Size* was randomized.

Instructions Given to Pilots

These are detailed in Appendix D.

Themes, Anecdotal Observations, and Dependent Variables (DV)

Themes. Formal qualitative analysis involves a search for *themes*—ideas central to the current subject, and usually expressed by more than one experimental participant (Miles & Huberman, 1984). Themes result from analysis of free-response text questions such as:

1. In your own words, describe what information we can get from a historical display such as NEXRAD.
2. What information can we get from a deterministic display?
3. What information can we get from a probabilistic display?

In qualitative analysis, we first look through participant responses, writing down statements that capture some discrete idea. If participants were responding to a specific question that had a "correct answer," we note both correct <u>and</u> incorrect statements they made. If, responses were more opinion-based, we treat opinions as ideas.

Once we have a list of these ideas, that list becomes our *rubric*, which is similar to an algorithm—specifications for how a procedure should be carried out. The rubric is then taken and reapplied to each participant's question responses, one by one, and a tally made of those ideas within the rubric that were found in each participant's responses.

Once all responses to all questions by all participants have thus been scored according to the rubric, the ideas most frequently expressed are classified as *themes*. But, we keep in mind that a theme does not have to necessarily be "correct;" it only has to be relatively common.

Anecdotal observations. Qualitative analysis can be confusing to some researchers, because it does not necessarily follow a neat, predetermined course set up and controlled by the researcher's own hand. Nonetheless, those who do conduct informal or semi-formal debriefing sessions after an experiment are often rewarded with valuable ideas and realizations offered up and/or stimulated by comments and answers from participants.

Dependent Variables. Strictly speaking, dependent variables usually result from quantitative analysis, being experimental outcomes whose numerical value we hypothesize will depend on the values we assign to our independent variables. As stated earlier, our sponsors wanted to emphasize qualitative analysis. Therefore quantitative DVs were restricted to answering a small number of specific questions such as:

1. "How much did you like each type of display?" (i.e., **H** vs. **H+D** vs. **H+P**")
2. "How much mental effort did it take you to use each type of display?"
3. "How much training and practice would each type of display require you to get good at it?"
4. "In actual hazardous-weather flight, how safe could each display type keep you?"
5. "What level of risk would you figure acceptable to fly through on a probabilistic display?"

These were set up in Likert-scale format, so that pilots could rate them on a 1-7 scale. Appendix E shows detail.

RESULTS

Pilot Participants

Eighteen general aviation pilots were recruited from a local flight school, and paid $50 USD each for their participation. Table 2 summarizes demographics.

Table 2. Pilot demographics (*n*=18, except n_{age} = 17). TFH = total flight hours.

Student	0	**CFII**	6	**Age-mean**	27.0	**TFH-mean**	1068
Private pilot	18	**Commercial**	9	**Age-median**	22.0	**TFH-median**	290
Instrument-rated	13	**ATP**	2	**Age-SD**	11.3	**TFH-SD**	2375
CFI	10	**Multi-engine**	6	**TFH-max**	10065	**TFH-min**	76

Pre-test Instructions and Practice

The pre-test instructions presented to pilots are shown in Appendix D. These included both text and screenshots of the application's Setup and Evaluation pages. Pilots were then walked through three practice scenarios, one for each of the three information-depiction types (H, H+D, H+P). They were then offered additional practice, but none requested it.

Qualitative Results

Free-Response Debrief Questions

Appendix E shows the free-response questions (#12-15), repeated below for convenience. These were scored according to standard qualitative methods (Miles & Huberman, 1984). They were important because they probed the understanding that pilots had of the three different kinds of information display:

12. In your own words, describe what information we can get from a historical display such as NEXRAD. (For instance, what do the colors mean? What benefit does looping the images add?)

13. What information can we get from a deterministic display? (What do its colors mean? How is it the same as the historical display? What extra benefit does it give us?)

14. What information can we get from a probabilistic display? (What do its colors mean? How is it the same as the other two? What extra does it give us?)

15. What about any of the displays felt confusing to you (or you suspect would confuse other pilots)?

How We Arrived at These Particular Questions

The reader may rightfully ask why we did not just directly ask pilots to explain their decision-making processes after each individual weather scenario. The answer is that we were trying to penetrate pilots' deeper thought processes, and anticipated superficial, repetitive responses had we merely tried to be direct. "I saw some weather and tried to avoid it" was the anticipated standard response, which would have been of little use in subsequent analysis.

Therefore, we instead chose to ask the abovelisted smaller number of relatively unstructured questions, to better assess the actual level of understanding pilots seemed to have about the three different kinds of weather-information displays. Question 15 asked pilots to tell us what, if anything, they felt was confusing about the displays. Such observations could be criticized as anecdotal, but they would give us a variety of pilot observations, likes and dislikes, all of which are important in an exploratory study such as this.

Scoring Issues

Any study using qualitative analysis must address methodological issues such as rater reliability, and implicit-but-unexpressed content.

Rater reliability. The process of scoring textual responses to questions can easily pose problems, a major reason why researchers so often shun qualitative analysis. The primary problem is that, even with a clear list of content we are looking for, one scorer can judge a given response to possess that content while another scorer can disagree.

Implicit-but-unexpressed content. A good example of the kinds of problems that vex qualitative analysis is trying to score implicit-but-unexpressed content. For instance, suppose a respondent writes that a Historical display "shows where the weather has been." Are we then to infer that this person derives *trend information* from that display, as opposed to purely knowing that a patch of weather moved from one location to another?

In truth, this is a foggy issue because we do clearly detect and infer qualities like *apparent motion* and *inertia* after seeing images of a moving object. Picture a movie clip of a speeding car. Then stop the clip. Would you perceive the car abruptly, instantly coming to a stop? Few of us would, because that would violate the laws of physics, specifically those of inertia (Anstis & Ramachandran, 1987). Now, apply this logic to a clip of moving weather. Seeing a series of still weather images should also induce perception of inertia—moreover, not just for general, overall direction-of-travel, but also for growth or shrinkage of individual parts of the weather system, as well as possible rotation of parts of it. Because, by the laws of physics, systems of fluids can have linear, rotational, *and* radial inertia.

Therefore, exactly how should we score a response like "shows where the weather has been?" Should we assume that this particular respondent—unlike the rest of humanity—is incapable of perceiving apparent motion and inertia? Yet, nowhere in the written words is anything specifically mentioned about "trend." Therein lies that particular problem.

Interrater reliability. In an attempt to "smooth" the sometimes-stark differences between raters, quantitative analysis often relies upon having multiple raters score the same text, then calculating an interrater reliability coefficient to estimate the degree of agreement between them. Naturally, high agreement is assumed to be good.

Our problem is that these pilots quite often failed to explicitly write down key scoring words (e.g., failed to clearly specify "This display shows trend in linear, rotational, and radial motion.") In fact, we will shortly see that we were usually lucky if they simply wrote "This display shows trend," without specifying what kind of trend.

This high level of inherent response ambiguity militates against the use of multiple raters and calculation of interrater reliability. Instead, what our particular situation lobbies *for* is arguably the simple scoring of pilot responses by a single rater—the approach used here—accompanied by a stern caveat to treat results as exploratory, not definitive.

Historical Display

The question. "In your own words, describe what information we can get from a historical display such as NEXRAD." One pilot essentially gave a flippant (but insightful) response of "Not much." The remainder treated the question more seriously.

The rubric. Each participant was given one point for each of the following five response elements a historical display provides:

1. Storm's intensity history/trend
2. Storm's size and/or shape history/trend
3. Storm's direction-of-travel history/trend
4. Storm's speed history/trend
5. Color scale represents storm intensity and/or strength of radar return

Results and interpretation. Figure 4 shows the frequency distribution of scores, "5" being the highest possible score a participant could get:

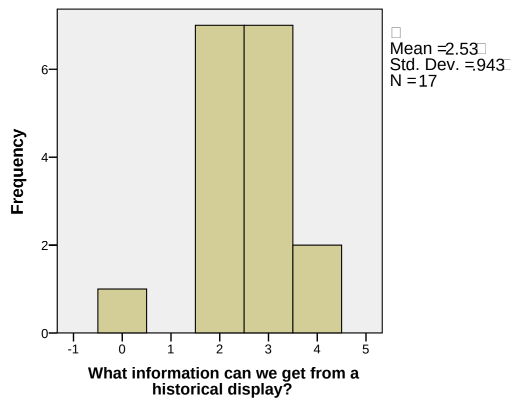


Figure 4. Frequency distribution of scores for Question 12.

Most pilots seemed to understand the basics. The highest-scoring pilots noted that storms can show more than one kind of trend, although no one noted all four kinds (intensity, size/shape, direction-of-travel, and speed).

It can be argued that most pilots probably would have correctly selected all the various forms of trend, had the question not been free-response, but rather had been explicitly listed, along with instructions to "Select all correct responses." This admittedly points out a fundamental weakness of qualitative analysis, namely that it may fail to completely uncover implicit knowledge.

Deterministic Display

The question. "What information can we get from a deterministic display?"

The rubric. One point each was given for each of the following eight response elements a deterministic display provides:

1. Storm's intensity history/trend
2. Storm's size/shape history/trend
3. Storm's direction-of-travel history/trend
4. Storm's speed history/trend
5. Color scale represents storm intensity and/or strength of radar return
6. Forecasts future weather
7. Is based on only one storm forecast model
8. Is unlikely to be highly reliable

Results and interpretation. Figure 5 shows the distribution of score results, with "8" being the highest possible score:

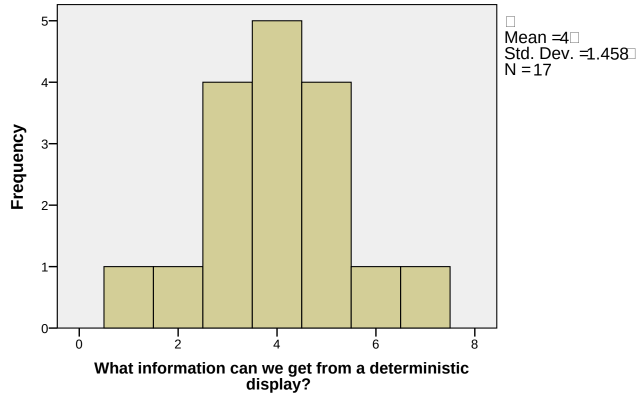


Figure 5. Frequency distribution of scores for Question 13.

Given the rather modest mean of 4, it may have superficially appeared as though many pilots failed to understand deterministic forecasting. That could well be.

Nonetheless, an alternative hypothesis needs to be considered: Pilots may merely have interpreted this second free-response question as being "What additional information can we get from a deterministic display?" Many apparently thought it unnecessary to include statements about trend that they had first expressed in the previous question about the historical display.

This points out a second weakness of qualitative methods, namely that questions have to be very carefully crafted and, ideally, a preliminary study (such as this one) should be made to look for unexpected results such as those that this question revealed.

One pilot stated the meanings of deterministic and probabilistic backwards, stating that the deterministic display was "taken from several models." This should not be taken to indicate widespread confusion amongst these participants over the deterministic-vs.-probabilistic distinction. But, given the hoped-for clarity of the pretest instructions (Appendix D), it does again point out the need for careful, in-depth training of basic concepts.

Probabilistic Display

The question. "What information can we get from a probabilistic display?"

The rubric. One point each was given for each of the following 10 response elements a probabilistic display provides:

1. Information about past convective weather and/or radar reflectivity
2. Storm's intensity history/trend
3. Storm's size history/trend
4. Storm's direction-of-travel history/trend
5. Storm's speed history/trend
6. H colors represent storm intensity
7. P colors represent % chance of encountering >40 dBZ weather
8. Forecasts future weather
9. Is based on many storm forecast model runs
10. Is more likely to be reliable

Results and interpretation. Figure 6 shows the distribution of score results, with "10" being the highest possible score:

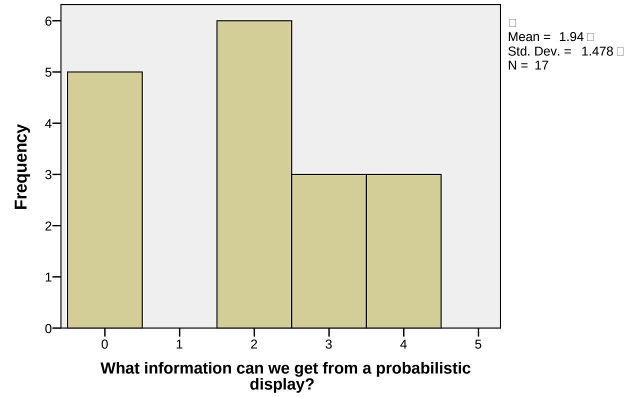


Figure 6. Frequency distribution of scores for Question 14.

Once again, the relatively low scores gave the impression that many pilots interpreted this question as being "What <u>additional</u> information can we get from a deterministic display?" This limited the usefulness of the question, and informed us that future attempts at qualitative analysis may require a preliminary study to help craft the questions we ask.

Summary Table

Table 3 summarizes the qualitative results. At least one important conclusion became clear only after organizing all answers into such a summary.

Table 3. Summary table of pilot responses to the three free-response questions.

In your own words, describe what information we can get from a historical weather display such as NEXRAD? (For instance, what do the colors mean? What benefit does looping images add?)	Rubric H-- Colors represent intensity	Rubric H-- Shows hx/trend in weather intensity/radar return	Rubric H-- Shows hx/trend in weather size/shape	Rubric H-- Shows hx/trend in wx direction of travel	Rubric H-- Shows hx/trend in weather speed	Rubric H--Can be used as a (poor) maneuver-planning tool	Rubric H-- Total score
Totals	16	14	2	11	0	0	43

What information can we get from a deterministic display? (What do its colors mean? How is it the same as the historical display? What extra benefit does it give us?)	Rubric H-- Colors represent intensity	Rubric H-- Shows hx/trend in weather intensity/radar return	Rubric H-- Shows hx/trend in weather size/shape	Rubric H-- Shows hx/trend in wx direction of travel	Rubric H-- Shows hx/trend in weather speed	Rubric D-- Predictive colors also represent intensity	Rubric D-- Forecasts future weather	Rubric D- Based on only one wx model	Rubric D-- Is unlikely to be highly reliable	Rubric D--Accuracy decreases as lookahead increases	Rubric D--Can be used as a maneuver-planning tool	Rubric D-- Total score
	15	10	2	10	1	4	17	**4**	**3**	**1**	1	68

What information can we get from a probabilistic display? (What do its colors mean? How is it the same as the historical display? What extra benefit does it give us?)	Rubric H-- Colors represent probability	Rubric H-- Shows hx/trend in weather intensity/radar return	Rubric H-- Shows hx/trend in weather size/shape	Rubric H-- Shows hx/trend in wx direction of travel	Rubric H-- Shows hx/trend in weather speed	Rubric P-- Predictive colors represent probability of >40 dBZ	Rubric P-- Forecasts future weather	Rubric P- Based on agreement of many wx models	Rubric P--Accuracy decreases as lookahead increases	Rubric P-- Is more likely to be reliable	Rubric P--Can be used as a maneuver-planning tool	Rubric P-- Total score
	6	5	2	3	0	5	6	**2**	**0**	**1**	3	33

Notice the six green-colored tally boxes at the bottom of Table 3. These green boxes show that only a handful of pilots offered detailed information about the major underlying differences between deterministic and probabilistic weather information. Was this merely oversight, or did most pilots truly not understand these differences and their importance? This is an important issue, and will receive further attention in the **Quantitative Results** section below.

Pilots' Observations and Comments

Table 4 organizes Question 15, pilots' anecdotal criticisms, observations, and comments about the three display types. Comments about the Historical-only display are shaded light green, those about the Historical+Deterministic display are shaded light yellow, and those about the Historical+Probabilistic display are shaded light blue.

Table 4. "What about any of the displays felt confusing to you (or you suspect would confuse other pilots)?"

1."The historical only had two loops so it wasn't very helpful so using this (H) model I was very risk averted [sic]."	2. "I think the historic can be confusing and miss leading because pilots can think this is where the storm is but in reality it's closing in on you and cause you to get in a storm."	3. "Historical was a pain to use because I had to predict where the weather was going to do anyways to make a decision. Both deterministic and probabilistic helped a lot."		4. "The deterministic display's "future" loop could be confusing because some pilots might put too much stock in its prediction of where the storm will be, due to its similarity in appearance to the "past" loop. Some kind of prominent disclaimer that the loop is occurring in the future and may not be very accurate might be helpful."	5. "The deterministic model might be more beneficial if it too used a different color scheme to illustrate predicted storm behavior. This I feel could prevent newer users from assuming future predictions are actual radar returns."	
6. "Colors on probabilistic was easy to confuse with an echo. Once a higher intensity spot popped up, then I remembered it had a higher probability."	7. "Probabilistic - we've been somewhat conditioned to think that light colors = less storm, dark colors = more storm. The P model shows a probability, not an intensity. As long as pilots understand that, no issue, but ensure that they do. That model is not the actual size of the storm, but where we think a certain "type" of the storm may be."	8. "I would like to know that the display takes into account confidence in the model using the display. This could be factored in by expanding the area of high probability where confidence is low and shrinking the area when confidence is high."	9. "To me, this was straightforward. I imagine the probabilistic display will confuse people because it switches from historical to probability but it is easy to grasp."	10. "I think the probabilistic display could be the most confusing to other pilots. Maybe the older pilot who isn't as technologically advanced would have trouble understanding the display."	11. "I felt the probabilistic display caught me off guard initially when it display an area had low dBZ but then progressively swapping to orange and even red forcing me to redirect late."	12."The probabilistic display was different but only took a few minutes to grasp conceptually. Love both deterministic and probabilistic and would consult both in making decisions (because both include NEXRAD and provide 2 different viewpoints for forecasted Wx)."

Historical. One pilot commented that "The historical only had two loops" (meaning "two frames"), which was technically incorrect (H-only had three frames). But, the point was probably that he/she would have preferred more historical frames. This is certainly worth considering, and would not be expensive to enact. Perhaps giving 5-minute updates, rather than the 10-minute updates these pilots saw, is also a possibility.

Two other pilots noted that the Historical-only condition (equivalent to current NEXRAD) felt inadequate compared to the predictive displays. That was also a point expressed off the record by at least a third of all pilots.

Deterministic. Two pilots (Table 4, yellow, comments 4, 5) noted that the H+D color scheme as we had it—using the same colors for historical and predictive frames—was potentially confusing. This was certainly a valid point, well worth considering, and contrasted with the H+P color scheme discussed next.

Probabilistic. Informally (by spoken word), nearly all pilots told us that they preferred the H+P display over the other two. Seven had various comments to make (Table 4, blue). These ranged from the completely noncritical (comment 12) to cautionary notes that not all pilots might understand the probabilistic display (comments 10, 11). One (comment 7) astutely noted that "the [probabilistic] model is not the actual size of the storm" (and, by inference, that could confuse people), and another (comment 6) concurred. Finally, one (comment 8) suggested we find a way to assign a confidence rating to each probability band. We will return to that idea later.

Should the Historical Information be Included Along with the Deterministic/Probabilistic Information?

Unfortunately, this was not included as a separate question. In retrospect, it should have been, since about ¾ of the dozen or so pilots we did query about this gave very firm positive responses that, yes, in their opinion, the Historical information was essential to giving a sense of trend—of "historical inertia" in terms of the storm's direction of motion and growth characteristics.

Are 10-Minute Updates Sufficient?

Informal polling of pilots during debrief revealed that about 60% thought they could get by with 10-minute weather-information updates. However, nearly all of those also stated that more-frequent updates (e.g., every 5 minutes) would be more welcome. This is sensible, given that an aircraft travelling at 120 kt could travel as far as 20 nm or more in 10 minutes, depending on winds aloft. In future studies, we recommend that preferred update rate be posed as a direct question.

Quantitative Results

Although our emphasis here was qualitative, a few quantitative measures were taken, particularly ones that could shed additional light on decision making and pilot acceptance of the technology.

Analysis of Free-Response Questions

A quick look at the bottom row of Table 3, the cells expressing total scores (H43, D68, P33), suggests that we could statistically analyze these totals on the three free-response questions. Table 5 shows the results of the non-parametric Friedman test for ranks. This implies significant differences between the number of responses given for the three information types.

Table 5. Free-response total scores for *What information can we get from a ___ display?* (N=17)

Display Type	**Descriptives**				**Friedman Test**			
	Mean	**SD**	**Min**	**Max**	**Mean rank**	**X^2**	**df**	**Sig.**
H	2.53	0.94	0	4	1.68	17.0	2	p =.0002
H+D	4.00	1.49	1	7	2.76			
H+P	1.94	1.48	0	4	1.56			
H-Historical, D-Deterministic, P-Probabilistic						**Bonferroni adj. p**		
Post-hoc comparisons (2-tailed)					**H vs. H+D**	.005		
					H vs. H+P	1.00		
					H+D vs. H+P	.001		

The post-hoc pairwise comparison tests indicate that pilots seemed to provide significantly more responses for the H+D (mean = 4.0) displays than they did for the H-only (mean = 2.53)) or the H+P (mean = 1.94) displays. However, this could merely be an artifact of the facts that: a) There was a lot more one *could* say about deterministic displays because they are more complex than historical displays, and; b) The question about deterministic displays *preceded* the one about probabilistic displays, so pilots may have simply not bothered to repeat answers about things that they felt were redundant. Future studies could therefore benefit from more careful wording of the questions to induce respondents to repeat elements of their previous answers, even if those were redundant (or, at least to begin later questions with a statement such as "everything I just said about historical displays, plus ___."

Analysis of Likert-Scale Questions

Results. Recall that the research design was set up as 3 × 3 = 9 treatments of *Weather Information × Storm Growth*, with weather-information display type being the primary interest. Likert-scale debrief questions were set up as a 1-7 rating scale, designed to be analyzed non-parametrically. Tables 6-14 show statistical results of a) the Friedman test, which estimates the chance of at least one of the three groups being reliably different from the other two, and b) *post-hoc* tests, which measure the three possible pairwise comparisons[4].

[4] SPSS applied the Bonferroni correction, which adjusted the significances of pairwise comparisons within each individual test to compensate for the number made (i.e., $p_{adj} = p_{raw}/3$).

Table 6. 1-7 Likert-scale ratings for *How much did you like each type of weather display?* (N=16)

Display Type	Descriptives: Mean	SD	Min	Max	Friedman Test: Mean rank	X^2	df	Sig.
H	3.50	1.32	2	7	1.68	24.4	2	p =.000005
H+D	5.38	0.62	4	6	2.76			
H+P	6.31	0.79	5	7	1.56			
H-Historical, D-Deterministic, P-Probabilistic						Bonferroni adj. p		
Post-hoc comparisons					H vs. H+D	.024		
					H vs. H+P	<.001		
					H+D vs. H+P	.102		

Table 7. 1-7 Likert-scale ratings for *How much mental effort did it take you to use each type of weather display?* (N=17)

Display Type	Descriptives: Mean	SD	Min	Max	Friedman Test: Mean rank	X^2	df	Sig.
H	3.82	1.74	1	7	1.74	2.22	2	p =.330
H+D	4.71	1.11	3	7	2.15			
H+P	4.65	1.54	2	7	2.12			

Table 8. 1-7 Likert-scale ratings for *How much training and practice would each type of display require for you to get really good at it?* (N=17)

Display Type	Descriptives: Mean	SD	Min	Max	Friedman Test: Mean rank	X^2	df	Sig.
H	4.35	1.69	2	7	2.03	1.22	2	p =.545
H+D	4.82	1.43	3	7	2.15			
H+P	4.47	1.66	2	7	1.82			

Table 9. 1-7 Likert-scale ratings for *In actual hazardous-weather flight, how safe could each display keep you?* (N=17)

Display Type	Descriptives: Mean	SD	Min	Max	Friedman Test: Mean rank	X^2	df	Sig.
H	2.82	1.07	1	5	1.00	32.1	2	p =1.07×10^{-7}
H+D	4.82	0.73	4	6	2.15			
H+P	5.71	0.92	4	7	2.85			
						Bonferroni adj. p		
Post-hoc comparisons					H vs. H+D	.002		
					H vs. H+P	<.001		
					H+D vs. H+P	.119		

Table 10. 1-7 Likert-scale ratings for *How much do you think others would like each type of weather display?* (N=17)

Display Type	Descriptives: Mean	SD	Min	Max	Friedman Test: Mean rank	X^2	df	Sig.
H	3.65	1.17	1	5	1.12	23.7	2	p =.000007
H+D	5.41	0.87	4	7	2.26			
H+P	5.94	0.90	4	7	2.62			
H-Historical, D-Deterministic, P-Probabilistic						Bonferroni adj. p		
Post-hoc comparisons					H vs. H+D	.002		
					H vs. H+P	<.001		
					H+D vs. H+P	.910		

Table 11. 1-7 Likert-scale ratings, *How much mental effort would it take other pilots to use each type of weather display?* (N=17)

Display Type	Descriptives: Mean	SD	Min	Max	Friedman Test: Mean rank	X^2	df	Sig.
H	4.06	1.56	1	7	1.85	1.47	2	p =.479
H+D	4.65	1.17	3	7	2.21			
H+P	4.41	1.58	2	7	1.94			

Table 12. 1-7 Likert-scale ratings for *How much training and practice would each type of display require for other pilots to get really good at it?* (N=17)

Display Type	Descriptives: Mean	SD	Min	Max	Friedman Test: Mean rank	X^2	df	Sig.
H	4.00	1.66	1	7	1.76	2.56	2	p =.278
H+D	4.76	1.35	3	7	2.24			
H+P	4.65	1.69	2	7	2.00			

Table 13. 1-7 Likert-scale ratings, *In actual hazardous-weather flight, how safe could each display keep other pilots?* (N=17)

Display Type	Descriptives: Mean	SD	Min	Max	Friedman Test: Mean rank	X^2	df	Sig.
H	3.59	1.06	1	5	1.09	29.4	2	p =$.4.1\times10^{-7}$
H+D	5.06	0.90	4	7	2.12			
H+P	5.82	1.02	4	7	2.79			
						Bonferroni adj. p		
Post-hoc comparisons					H vs. H+D	.008		
					H vs. H+P	<.001		
					H+D vs. H+P	.146		

Table 14. 1-7 Likert-scale ratings for *How helpful was the range ring in keeping you separated from the weather?* (N=17)

Display Type	Descriptives: Mean	SD	Min	Max	Friedman Test: Mean rank	X^2	df	Sig.
H	5.12	2.12	1	7	1.53	14.9	2	p =.001
H+D	6.24	0.90	4	7	2.12			
H+P	6.47	0.62	5	7	2.35			
						Bonferroni adj. p		
Post-hoc comparisons					H vs. H+D	.259		
					H vs. H+P	.049		
					H+D vs. H+P	1.000		

Interpretation. As a group, pilots liked the predictive displays significantly better than plain looping NEXRAD (Table 6), and thought other pilots would too (Table 10). They felt that all three display types took about the same amount of mental effort (Table 7) and would so for others (Table 11). They thought all three displays would take about the same amount of training for them to attain proficiency (Table 8), and for others as well (Table 12). Finally, they thought the predictive displays would keep them significantly safer in actual hazardous-weather flight than looping NEXRAD (Table 9), and would keep other pilots safer, too (Table 13).

Note that most of these Likert-scale results, rather than expressing distinct perceived superiority of the probabilistic forecast display, instead implicitly supported distinct perceived *in*feriority of the historical-only (looping NEXRAD) display. This is evident because H+D and H+P mean ratings were consistently significantly higher than H-only, but H+D vs. H+P means were not significantly different from one another (despite Ps being slightly higher on most comparisons).

What this tells us is that *any* forecasting capability showing predicted weather and aircraft positions was warmly welcomed by these pilots. These results were also consistent with subjective statements to that effect by many pilots during debrief. Any finer discrimination between the H+D and H+P depictions, however, will take further, focused study.

Sidebar. In future studies, it is probably unnecessary to ask pilots how they think other pilots will relate to technology such as this. Spearman *rho* correlations between Likert-scale ratings on the 12 "you" questions and each pilot's corresponding "other pilots" question rating ran from .36-.89, with the mean rho = .60, implying moderately strong correlation. Most of the *rho p*-values were significant, with only three greater than 05, the highest *p* being just .15. Mean shared variance (average rho^2) was .40, implying considerable overlap in how pilots viewed their own reactions to the displays, and how they imagined other pilots would react. Finally, 94.6% of all "you" ratings differed from their corresponding "other pilots" rating by one point or less.

The net effect implied that asking pilots what they thought about other pilots was largely redundant.

Analysis of IVs

Recall that our main IVs were :

A. *3 Weather Information Types*:
   1. Historical **H**
   2. Historical + Deterministic **H+D**
   3. Historical + Probabilistic **H+P**

B. *3 Storm Growth Types:*
   1. Getting smaller over time **SM**
   2. Static **ST**
   3. Getting larger over time **LA**

Sponsors expressly dictated that we *not* measure point-of-closest-approach-to-hazardous-weather (PCA). Nonetheless, one very simple, easily collectable/analyzable measure recorded by the PTS for each trial was the total scenario *Elapsed Time* (*ET*, time from start to finish of each individual weather scenario). ET could be loosely interpreted as a proxy for "mental effort," with larger ETs standing in for greater mental effort. We therefore extracted that ET information.

Repeated-measures ANOVA on scenario ET is summarized in Tables 15 and 16:

Table 15. Descriptives for ET Weather Information (WI), Storm Growth (SG)

WI	SG	**Mean ET** [A]	**SD**	**WI grp mean** [A]	**SG**	WI	**Mean ET** [A]	**SD**	**SG grp mean** [A]
H	SM	58.9	27.9	**60.5**	**SM**	H	58.9	27.9	75.4
	ST	57.5	31.0			D	85.0	42.2	
	LA	65.1	26.4			P	82.2	42.8	
D	SM	85.0	42.2	82.3	**ST**	H	57.5	31.0	68.5
	ST	69.9	28.2			D	69.9	28.2	
	LA	92.2	48.9			P	78.3	28.5	
P	SM	82.2	42.8	82.1	**LA**	H	65.1	26.4	81.0
	ST	78.3	28.5			D	92.2	48.9	
	LA	85.7	35.9			P	85.7	35.9	

[A] Elapsed time, in seconds.

Table 16. RM-ANOVA summary

IV	$p_{Mauchly}$	$p_{RM\text{-}ANOVA}$ [A]
Weather Info type	.509	.00043
Storm Growth type	.216	.076
WI × SG	.734	.754

[A] Sphericity assumed

The left side of Table 15 summarizes how, as a group, the historical-only (H) *Weather Information* depictions had significantly lower mean elapsed time (60.5 sec) than either the deterministic (D, 82.3 sec, p = .003) or probabilistic (P, 82.1 sec, p = .001) depictions. This was consistent with Table 7 earlier, where pilots directly rated H depictions as requiring the least mental effort.

Nonetheless, this probably merely means that historical-only depictions, while "faster" to use, were only faster because a) the loop ran faster (3 vs. 7 frames), and b) the task itself amounted to just *taking a quick guess* about where weather was going. It does not imply that H-only depictions resulted in better weather avoidance. In comparison, predictive displays a) took longer to loop, b) presented far more information to process, and c) afforded opportunities to test various maneuvers *before* moving forward in time. So, they likely traded speed for accuracy in terms of weather-avoidance performance. However, a definitive answer would require analyzing PCA-to-heavy weather, which sponsors expressly forbade.

Meanwhile, the right-hand side of Table 15 addresses the possible influence of changing storm size—particularly that storm growth might take the most cognitive effort to process. This was not supported. The results show that the three categories of *Storm Growth* type did not significantly differ in elapsed time. Only the static group (ST, 68.5 sec) showed trend for shorter ET with only the storm-growing-larger group (LA, mean ET = 81.0 sec, p = .072). If there were a reliable effect there, it would likely take a larger sample of pilots to demonstrate it.

Individual differences. While quantitative analysis of mean ranks, such as just described, is one important way of telling a story about data, it is not the only way. Description of individual differences—how pilots varied among themselves—can be helpful as well. This can suggest where we might find issues in areas like product acceptance or training.

For instance, Figure 7 shows wide variation in how much pilots said they liked the H-only display, with acceptance looking better for the forecasting displays (higher ratings + less variation). Figure 8 shows wide variation in perceived mental effort for all three display types, with H-only having the most variation. This would have implications for training, implying that (as we might expect), some pilots would find it easy to understand forecast displays, while others might find it difficult. Figure 9 reinforces Figure 8's implications, showing wide variation in perceived training needs for all three display types.

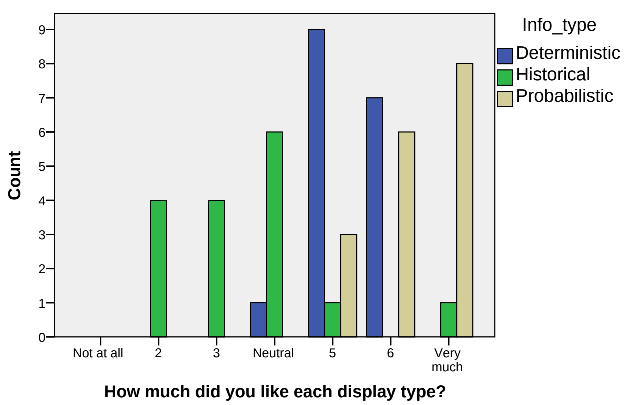


Figure 7.

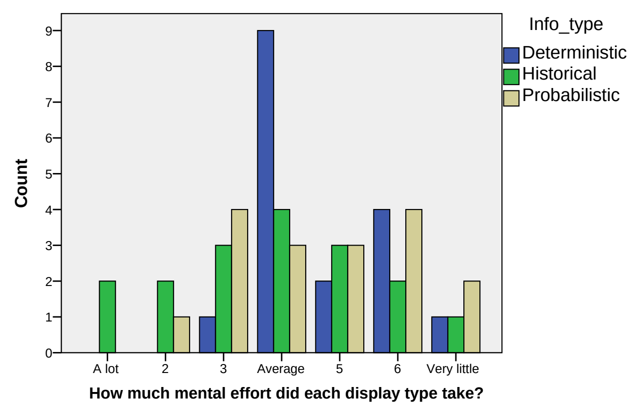


Figure 8.

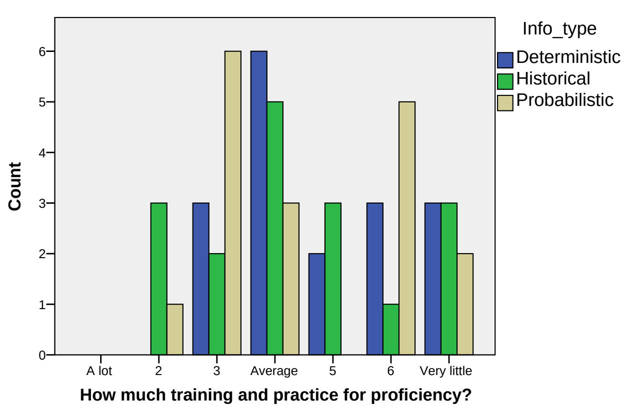


Figure 9.

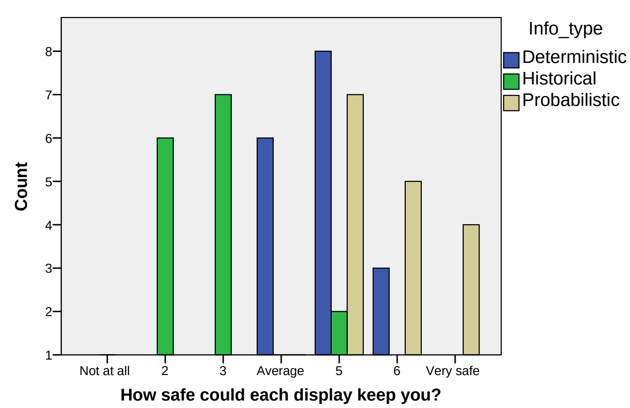


Figure 10.

Figure 10 clearly shows the H-only depiction perceived as least-safe. Table 9 verifies that, statistically. Also, we might be tempted to see the H+P display as perceived safest; however, Table 9's post-hoc H+D-versus-H+P test fails to verify (due, perhaps, to small sample size).

Certainly, what all these individual-difference breakouts show is the need for a repeat study with more pilots.

Acceptable Risk

One final topic of great interest, and having to do with individual differences, is the question of just how much stated risk-taking do pilots *think* is acceptable versus how much risk-taking is probably *actually* acceptable. Figure 11 summarizes how much risk our pilots said they, personally, would accept, which we told them should be calibrated as:

"0%"	meaning	"I wouldn't want to ever take any risk under any circumstances whatsoever,"
"50%"	meaning	"I'd accept a 50-50 chance of running into heavy weather," and
"100%"	meaning	"I'd completely disregard the weather display and fly anywhere, anytime."

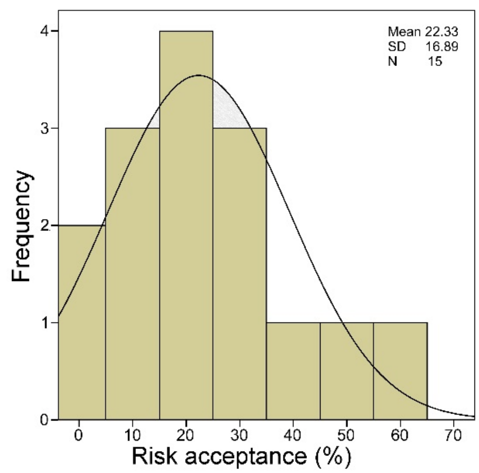


Figure 11. "What level of risk would you figure acceptable to fly through on a probabilistic display? ____%"

Heavy weather, again, means composite reflectivity > 40 dBZ.

Unfortunately, Figure 11 belies an issue of greater complexity than it may appear. What "risk" means, within the context of a *graphical depiction of weather risk*, turns out to be a far more vexing problem than one might imagine, as we are about to see.

What is Aviation Weather Risk?

How We Defined "Risk"

In this study we instructed pilots to consider probabilistic weather information as being *the percent chance of encountering worse than 40 dBZ ("heavy") radar reflectivity* in a given place at a given time. At first blush, that definition sounds clear enough, if we assume a simplified definition of "risk" merely being "the *chance* of some bad event happening," and leave out "*how* bad, or *how* expensive that event would be," which are features often technically included in calculations of risk.

The deeper issue is, though, if we look closer, armed with systematic mathematical reasoning, how confident should we truly be that even our simplified definition is both comprehensive and easy for all pilots to understand?

What Probabilistic Risk Really Means

Total volume under a curve, $0 \leq p \leq 1$. First, let's consider what our definition implicitly means. Does a probability region shown on a weather display *have* to mean "a *certainty* that *somewhere* within a given enclosed area we will find heavy weather?"

Not necessarily. Within an entire defined geographic region, heavy weather may or may not develop. And, in forecasting, we can assign any number between 0-100% to that total probability (0-1.0, if we use proportions). The only constraint is that the total volume under the curve be equal to some proportion greater than or equal to 0 and less than or equal to 1.0. Mathematically,

$$0 \leq p = \iint_{-\infty}^{\infty} f_{x,y} \leq 1.0 \qquad (1)$$

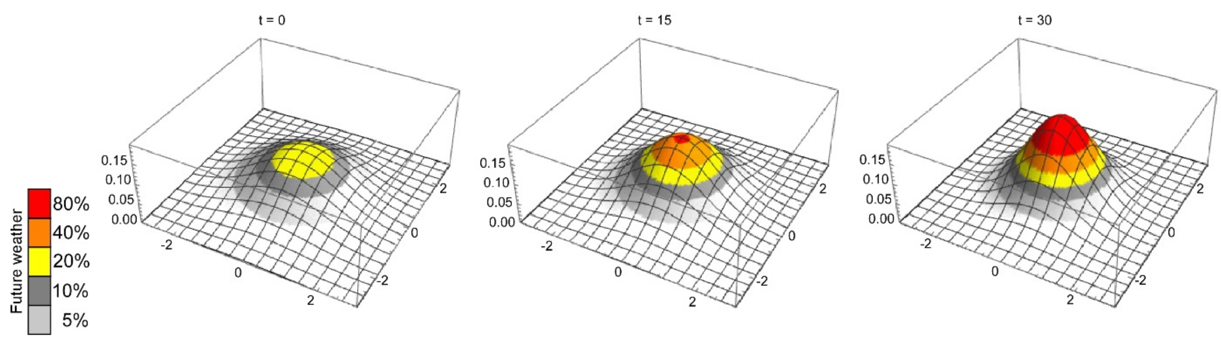
So, do pilots intuitively and effortlessly understand this kind of calculus?

The probability curve flattens, the farther out we forecast. This is something the ATSC (2013) study revealed, namely that a probabilistic forecast region can end up looking larger than a deterministic forecast region. The reason for this is that prediction error increases as lookahead increases. So, if error increases, that means we acknowledge more locations where heavy weather might occur. "More locations" means "a bigger geographic region, but with lower local probability estimates in each individual part of that that bigger overall region."

This presents a problem. This flattening of our probability curve means that *a probabilistic forecast region may look less dangerous, the farther the lookahead.* So, will pilots automatically and effortlessly understand this, too?

This is easier to visualize graphically. Figure 12 shows an idealized region, a 3D plot of Equation 2, a 2D Gaussian ("normal curve") probability density function (pdf), the volume underneath which conveniently always equals 1.0, no matter what the standard deviation (σ, sigma). The symbol *e* represents the natural logarithm. For our purposes, σ will represent prediction error, so the greater the forecast lookahead, the larger σ will be.

$$\frac{e^{\frac{-x^2}{2\sigma^2}}}{\sigma\sqrt{2\pi}} \frac{e^{\frac{-y^2}{2\sigma^2}}}{\sigma\sqrt{2\pi}} \qquad (2)$$



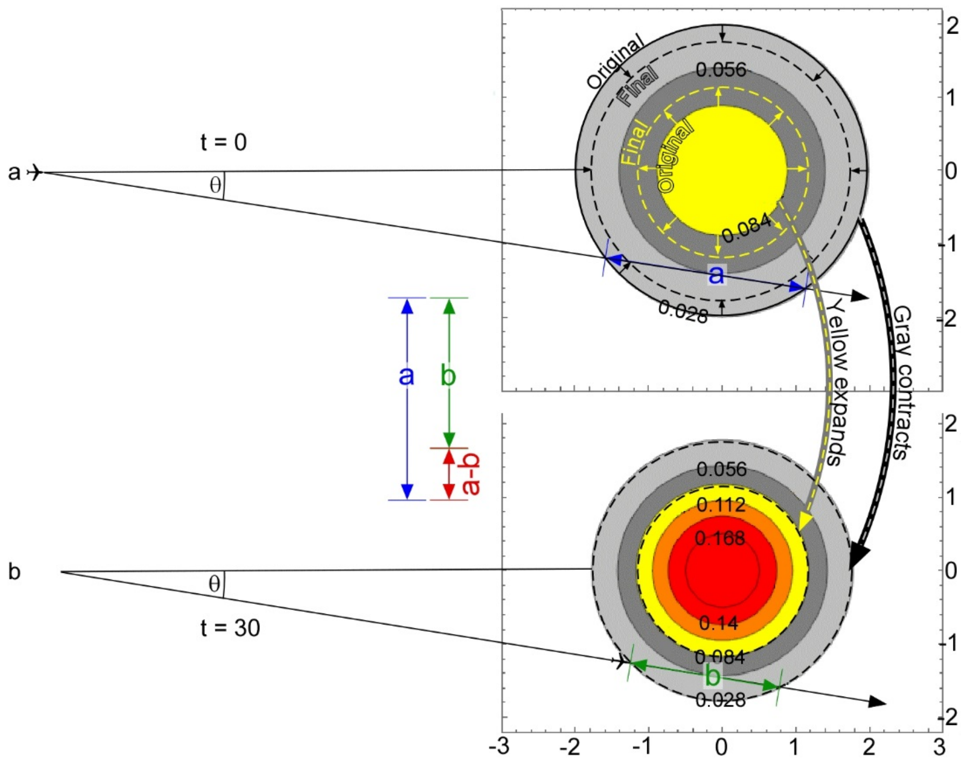
a → b → c

Figure 12. A 2D Gaussian probability density function representing, a) an idealized "weather cell" far from the aircraft, having prediction error σ = 1.20, b) as the aircraft approaches, σ = 1.05, c) getting very close σ = .90. The progression a→b→c shows the pdf peaking as forecast lookahead decreases and prediction error decreases. Fig. 13 elaborates.

Now, IF a pilot saw the yellow region in Figure 12a, and thought that was an acceptable level of risk, and flew toward that yellow region, as he/she approached it, time would have passed, prediction error would have decreased, and what was once yellow might now look red, as Figure 12c shows.

Risk is also a function of exposure duration and instantaneous risk. The changing shape of the pdf has other complexifying effects, in that it can change the amount of time the pilot is exposed to weather risk. Figure 13 focuses on just 12a and c, showing the effect that lookahead and prediction error have on both the exterior and interior of the pdf. The light gray area—the exterior size of the pdf—contracts as lookahead decreases, while the internal size—the original yellow area—expands, the middle turning to red.

In this particular example of heading θ, total eventual exposure time *decreases*, as we can see by comparing the lengths of segments *a* versus *b*. But, if θ had been smaller, risk would have *increased* because the path would have taken us through higher-risk areas. In fact, this points out an even-greater difficulty of trying to estimate overall risk when risk itself is changing from one moment to the next.



a b

Figure 13. Top-down views of Figs. 11a and c. a) Aircraft start position at time t = 0, b) Aircraft position at t = 30 minutes. Note the light-gray region contracting while the yellow region yellow region expands. In this particular example with an initial deviation heading θ, an aircraft intending to just graze the dark gray zone ends up experiencing shorter risk-exposure duration (length of green segment "b") than originally planned (length of blue segment "a").

Additional Complicating Factors in Total Risk Calculation

We can list additional factors making use of probabilistic weather displays difficult, roughly rank-ordered in obviousness from most to least. Take note that we will purposely exclude some pilot- and aircraft-related factors (e.g., pilot skill and aircraft's ability to handle severe weather) simply because they are not directly related to our main focus on visual displays of weather information.

Radar does not always reveal true danger. First, radar reflectivity is only a proxy for weather risk, and an imperfect one at that. While we can use different warning colors, what those colors actually represent in a probabilistic display is the probability of encountering > 40 dBZ composite reflectivity in a given region at a given time. Factors like turbulence and wind shear are currently not represented in NEXRAD.

Real storms are irregular. Second, real storms rarely look, behave, or occupy easily predictable locations the way our idealized pdfs of Figure 12 and 12 look. Those irregularities will increase complexity and decrease predictability.

Sparse update rates disguise change. Third, at least for the next decade or so, our display update rates will probably remain on the order of every 5 minutes at best. Naturally, sparse updates cannot give us fine temporal detail about rapidly changing storm shape, intensity, or location.

This non-obviousness creates a serious human-factors concern. Because, given our use of color to represent instantaneous local probabilities, *a long-lookahead forecast will tend to look much less dangerous than a short-lookahead forecast*. A quick glance at Figure 13a versus 12b illustrates this.

Will training be required? Taken together, the list of issues just described will make net risk prediction along any given route difficult through the kinds of complicated probability fields we know will result from actual weather. If it takes such deep thought and sophisticated mathematical visualization software to clarify these issues, it is hard to imagine that training would *not* be required.

Can technology help? In theory, risk estimation and safe course-plotting could both be greatly simplified by giving pilots the abilities to

1. display looping frames,
2. "scroll back and forth in time,"
3. stop at any given frame, and to
4. check PCA to a given probability level by use of a range ring.

These are display enhancements that should be given serious investigation by researchers.

DISCUSSION

The graphical depiction of probabilistic weather information is assumed by many to represent a significant advancement in weather technology in the cockpit. Probabilistic displays would depict the degree of agreement between multiple weather forecasts, giving pilots estimate of future risk associated with local geographical areas. It is therefore indisputable that probabilistic displays inherently contain more information, and would be generally more reliable than deterministic forecasts, which depict only the results of a single forecast model run one, single time.

However, such advanced displays raise serious human-factors and aviation psychology issues. What is the minimum amount of information pilots need to see to stay safely separated from adverse weather? How well would pilots understand the displays? What training would be necessary? How would they affect workload? What net effect would they ultimately be likely to have on safety during adverse-weather flight?

In search of answers to these questions, the present study beta-tested a simple part-task simulation (PTS) of looping probabilistic weather information. Eighteen general aviation pilots were shown three types of weather-information depiction: Historical-only (H-only), Historical + Deterministic (H+D), Historical + Probabilistic (H+P), within the context of three types of storm-cell evolution (Storm size decreasing, Storm size static, Storm size increasing), resulting in nine weather scenarios shown to each pilot. Several behavioral-response measures were collected, and post-experimental questions asked of pilots in order to gain insight into their decision making.

Empirical Results

As a group, 18 pilots

1) reported that they liked the H+P display most of the three types, and significantly preferred either kind of predictive display to the H-only display (Table 6, and Fig. 7)
2) felt that all three display types would take about the same mental effort to use (Table 7), and about the same training and practice to master (Table 8). However, individuals varied greatly in these opinions (Figs. 7, 8).
3) felt the H+P display would keep them safest during adverse-weather flight, and that the two predictive displays would both be significantly safer than the H-only display (Table 9, Fig. 10).
4) spent significantly less time maneuver-planning using the H-only display (Tables 15, 16), most likely because it was simplest, had the shortest looping times, and afforded only "guesstimation" for maneuver-testing rather than affording opportunities for detailed maneuver-testing before initiation.
5) reported they would choose to fly through an average of 22.33% risk of encountering heavy weather on a probabilistic weather display, but there was wide variation among individuals (Fig. 11).
6) spent the least time maneuver-planning on static-sized storms, with trend ($p = .076$) for increased time when storms either shrunk or grew over time (Tables 15, 16).

Results From Informal Polling and Pilot Comments

1) Most pilots said they could function with 10-minute updates, but 5-minute updates were preferable.
2) Most said they prefer at least 30 minutes historical lookbehind and about 45 minutes forecast lookahead in order to navigate past a storm system 40 nm wide.

3) Most said that a simplified, NEXRAD-like color scheme appears appropriate for historical weather.
4) Several pilots reported that a strikingly different color scheme (e.g., the CIWS scheme) is needed to clearly differentiate between historical and forecast weather information.
5) A looping format helps impart a sense of weather direction and storm-size/shape changes.
6) Several mentioned that the ability to "scroll back and forth in time" and stop at any given frame, would be extremely helpful in estimating point-of-closest-approach (PCA) to hazard.
7) Most agreed that a range ring around the aircraft icon will likely prove essential to accurately estimating point-of-closest-approach to hazardous weather.
8) It may be worthwhile to develop an entirely, radically new way of graphically representing probability for pilots.

Results Derived From Deduction

1) Because of prediction error, the farther forward in time we look, the flatter the probability density functions (pdfs) generated by probabilistic weather displays will tend to be (Figs. 11, 12). If these are color-coded, it may give pilots the impression that distant convective weather is safer than it actually is.
2) Because actual risk calculation involves many complicated factors (Eq. 2), given only a looping probabilistic display, pilots will probably "guesstimate" the risk of any given flightpath by use of some kind of heuristic.
3) However, given a probabilistic display with
   a) the ability to loop frames, scroll, and stop at any given frame,
   b) a 20-nm range ring around the aircraft icon,
   c) a weather update every 5 minutes, and
   d) sufficient training and simulator practice,

   pilots should theoretically be able to reliably plan safe maneuvers around heavy weather even when the weather systems are irregular in shape and rapidly changing.

Conclusions

Probabilistic information displays are non-trivial, and it would be a mistake to assume that pilots can quickly learn to interpret and use them effectively. Whatever scheme is used to represent probability will be subject to underlying characteristics far more complicated than meet the untrained eye. Unlike most meteorologists touting probabilistic displays, most pilots do not have formal training in probability theory or application. It is human nature for most of us, pilots and non-pilots alike, to believe we understand complex phenomena better than we actually do.

SUGGESTIONS FOR FUTURE RESEARCH

Further research is imperative in order to inform policy makers. Studies should

1. employ greater numbers of pilots, to give increased statistical reliability,
2. explore what kind of training would help pilots understand the issues raised in the current study,
3. continue to use weather displays having a range ring, and capable of looping and "time-scrolling,"
4. test storms that rapidly change shape, size, and intensity,
5. measure final point-of-closest-approach to eventual actual weather, in order to have objective measure of pilot safety performance.

Additionally, coordination is needed with professional groups doing actual probabilistic weather modeling, to ensure that the experimental stimuli used in our studies will accurately reflect the real-world levels of performance these forecasters can achieve.

ACKNOWLEDGMENTS

We are grateful to Kevin Gildea, Katrina Avers, and our sponsors Eldridge Frazier and Gary Pokodner for reviewing this manuscript and providing helpful comments. Special thanks for recruitment of pilot participants go to David Lodes, Chief Flight Instructor, and Kenneth Carson, Program Director of the University of Oklahoma Department of Aviation, and their staff. Finally, we thank Suzanne Thomas for assistance with data collection.

This research is in response to requirements and funding by the Federal Aviation Administration (FAA). The views expressed are those of the authors and do not necessarily represent the official policy or position of the FAA.
FAA Contract: DTFAWA-15-D-00036
Task Order Number and Task Name: Wx 1, FY18
Deliverable: Analysis Report of Demonstration Results of Prototype Tool Enhancement
Delivery Status: Final

REFERENCES

Ames, D.R. & Kammrath, L.K. (2004). Mind-reading and metacognition: Narcissism, not actual competence, predicts self-estimated ability. *Journal of Nonverbal Behavior*, 28(3), 187–209.

Anstis, S., & Ramachandran, V.S. (1987). Visual inertia in apparent motion. *Vision Research, 27*(5), 755-764.

Atmospheric Technologies Service Corporation. (2013). Final report: Demonstration comparing the effects of probabilistic and deterministic forecast guidance on pilot decision-making and performance (Report FAA 20131210.1). Norman, OK, ATSC.

Brewster, K. (2016). Personal conversation. Associate Director, Center for Analysis and Prediction of Storms, Norman, OK.

Evans, J.E., Carusone, K., Wolfson, M. Crowe, B., Meyer, D., and Klingle-Wilson, D. (2002). The Corridor Integrated Weather System (CIWS). In *10th Conference on Aviation, Range, and Aerospace Meteorology*.

Federal Aviation Administration. (2013). *Advisory Circular AC 00-24c*. Downloaded Dec. 11, 2018 from https://www.faa.gov/documentlibrary/media/advisory_circular/ac%2000-24c.pdf.

Guinn, T.A., & Barry, R.J. (2012). The certainty of uncertainty. Understanding and exploiting probability-based aviation weather products. *Journal of Aviation/Aerospace Education & Research, 21*(2), 59-78.

Kahneman, D., Slovic, P., & Tversky, A. (Eds.). *Judgment under uncertainty: Heuristics and biases*. Cambridge University Press.

Knecht, W.R. (2016). Current NEXRAD cannot reliably enable safe flight around heavy weather. *The International Journal of Aviation Psychology, 26*(1-2), 46-61.

Kruger, J. & Dunning, D. (1999). Unskilled and unaware of it: How difficulties in recognizing one's own incompetence lead to inflated self-assessments. *Journal of Personality and Social Psychology*, 77(6), 1121–1134.

Morss, R.E., Demuth, J.L., & Lazo, J.K. (2008). Communicating uncertainty in weather forecasts: A survey of the U.S. public. *Weather and Forecasting, 23*, 974-988.

National Transportation Safety Board (2005). *Risk factors associated with weather-related general aviation accidents* (Safety Study NTSB/SS-05/01). Washington, DC: Author.

National Weather Service. (2018). *FAQ – What is the meaning of PoP*. Downloaded Dec. 10, 2018 from https://www.weather.gov/ffc/pop.

Mather, G., & Challinor, K.L. (2009). Psychophysical properties of two-stroke apparent motion. *Journal of Vision*, 9(1), 1-6.

Miles, M.B. & Huberman, A.M. (1984). *Qualitative data analysis: A sourcebook of new methods*. London. Sage Publications.

Wolfram Research, Inc. (2018). *Mathematica, Version 11.3*, Champaign, IL.

Wu, S-C, Luna, R., & Johnson, W.W. (2013). Flight deck weather avoidance decision support: Implementation and evaluation. In *Digital Avionics Systems Conference* (DASC), 2013 IEEE/AIAA 32nd (pp. 5A2-1). IEEE.

APPENDIX A

The Weather Generator: Generating and Sequencing Weather Images

Underlying Logic

This Appendix details the logic and method guiding the Weather Generator program, which created our weather stimuli. This essentially centered on generating *keyframes*, and then morphing between them. Wikipedia (2018) provides a concise definition: "A keyframe in animation and filmmaking is a drawing that defines the starting and ending points of any smooth transition."

Generating Individual Keyframes

Modeling NEXRAD imagery. Weather is chaotic, in the formal mathematical sense. Namely, it involves coupled systems whose state at time $t+1$ is a function of the system's state at time t. Specifically, weather is conceptualized as a complex 3D coupled system where unit volumes approaching zero embody variables such as temperature, pressure, humidity, and direction of movement, and these volumes share information with nearest-neighbor volumes to create large, dynamical systems such as storms.

The current state of atmospheric modeling employs supercomputers calculating complicated, coupled differential equations (equations that compute the state of a system at time t, and then propagate the results between nearest neighbors to generate a new state of the system at time $t+dt$, with dt being "delta-t," the time difference, or *differential*).

We had no supercomputer, nor wish to employ one, since this project's needs were far simpler than those of the NWS. We only needed to generate images reasonably similar-looking to the kinds of storm cells we commonly see in NEXRAD.

To do that, we utilized a computer algorithm commonly used by video game developers to generate terrain maps for games. The *diamond-square algorithm* can create square, fractal heightmaps—literally, "maps of heights," in computer memory, created by a a recursive mathematical algorithm (a computer function or procedure that repeatedly calls itself, in this case, dividing a fixed-size square array of memory slots, representing pixels, into increasingly smaller squares, which are all mathematically related to one another)—such as shown in Figure 14. Code is detailed in Appendix B.

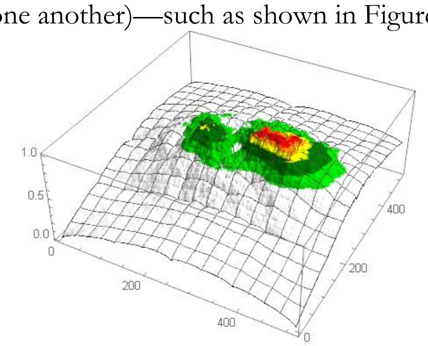


Figure 14. A 513×513-pixel fractal heightmap created in *Mathematica* by the diamond-square algorithm. A squashing function tapers edges of the heightmap while a color function assigns color to various ranges of height.

Tapering heightmap edges. A close look at Figure 14 reveals that it tapers off at the edges. However, raw heighmaps do not normally taper off, and so would not look much like storm cells. To correct that, we multiplied the height of each value in the heightmap by a 2D *squashing function*—a sigmoid, or S-shaped function with lower bound 0 and upper bound 1, centered at array position 256×256. This had the effect of "squashing" height values more severely the farther away from array center the given value.

$$squash(r,c) = \frac{4}{\left(\left(1+2^{w_r\left|r-\frac{size}{2}\right|}\right)\right)} \frac{4}{\left(\left(1+2^{w_c\left|c-\frac{size}{2}\right|}\right)\right)} \quad (3)$$

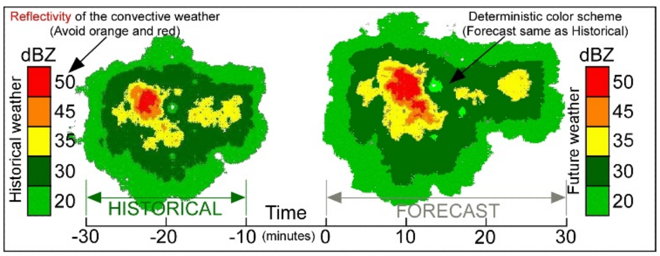
where r = row, c = column, w_r = row slope, w_c = column slope, and *size* = size of the square array (here = 513).

Normalizing the array. Next, we normalized each heightmap by forcing all values $h_{x,y}$ to lie between 0 and 1 ($0 \le h_{x,y} \le 1$). This involved finding the smallest (*min*) and largest (*max*) values in the array, and then multiplying all values by $(h_{x,y} – min)*(1/(max-min)$. This, of course, had the result of reducing *min* to 0, elevating *max* to 1, and adjusting everything in between accordingly to a “normal” range 0-1.

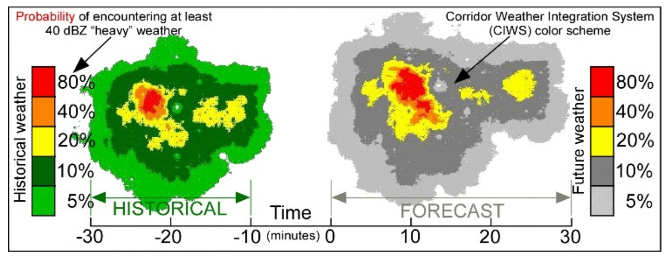
Adding color. To add NEXRAD-like color to each pixel, we used two *color functions*. These translated different ranges of height into five specific colors. The following is an example:

```
cf = Function[z,
  If[z < 0.70, RGBColor[1, 1, 1],              (* white *)
    If[z < 0.79, RGBColor[0, 0.75, 0],         (* med. green *)
      If[z < 0.88, RGBColor[0, 0.4, 0],        (* dark green *)
        If[z < 0.92, RGBColor[1, 1, 0],        (* yellow *)
          If[z < 0.95, RGBColor[1, 0.5, 0],    (* orange, the color pilots must avoid *)
            RGBColor[1, 0, 0] ]]]]]];          (* red *)
```

The first color function represented standard NEXRAD reflectivity in decibels (dBZ, Fig. 15a). The second color function emulated the Corridor Integrated Weather System (CIWS, Fig. 15b) color scheme, but now representing *probability*. Specifically, color here represented *the percent chance of this region containing heavy weather* (with “heavy weather” denoted by the FAA standard, i.e., “ > 40 dBZ WSR-88D composite reflectivity”)



a



b

Figure 15. a) a simple 5-color, NEXRAD-like scheme for historical display and deterministic forecast, b) the probabilistic color scheme, showing standard NEXRAD-like historical weather, but now with a CIWS-like probabilistic forecast.

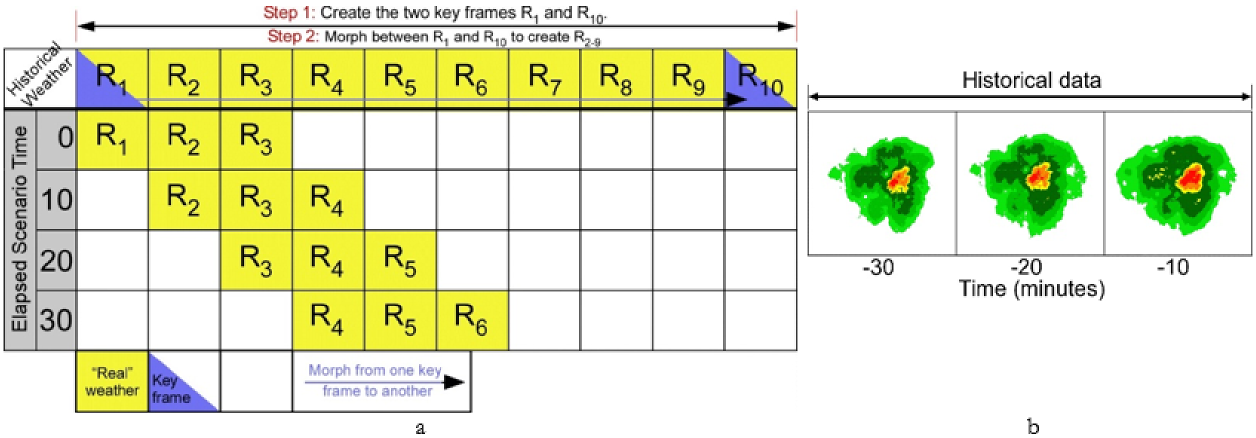
Generating Sequences of Frames

In this experiment we tested three basic types of weather-information display

1. 3 frames Historical only
2. 3 frames Historical followed by 4 frames Deterministic forecast
3. 3 frames Historical followed by 4 frames Probabilistic forecast

Each frame represented weather information sampled at 10-minute intervals. Sequences of frames consisted of one keyframe at either end of the sequence, plus morphed images in between.

Historical weather. Think of “historical” frames as representing “real weather,” or “weather that ‘really’ happened or will ‘really’ happen.” These “real” frames were given the designation “R” in Figure 16a, which shows the sequencing chart for historical-only weather.



a b

Figure 16. a) the sequencing chart for historical-only weather, b) an example of frames R_1, R_2, and R_3.

To interpret Figure 16, think of Elapsed Scenario Time zero (EST_0) as the exact time when a pilot starts watching the weather display. At EST_0, three frames were shown in a loop—R_1, R_2, and R_3. The pilot then had the opportunity to make a maneuver to avoid the storm cell. After the maneuver, he/she hit a button to "advance the viewer 10 minutes in time," as if the aircraft were flying along that heading, at that speed, for 10 minutes. Next, at EST_{10}, frames R_2, R_3, and R_4. were shown, the pilot had another chance at maneuver, and so forth.

In this first experiment, only three opportunities were given for maneuver—the first at EST_0, the second at EST_{10}, and the third at EST_{20}. This three-maneuver limit was imposed in order to motivate pilots to choose each maneuver wisely. This limit may change in future experiments.

In Figure 17, notice that the first row contains 10 "R" frames, rather than just the six that were actually displayed. The reason for having 10 was simply to plan ahead for future studies when we may be called upon to estimate point-of-closest-approach (PCA) to the storm cell. Frames R_7-R_{10} will then be available to know where the 'real weather' was as the aircraft advanced an additional four frames (40 minutes).

Forecast weather. Forecast frames are meant to represent "NWS-predicted weather." Figure 17 shows the sequencing chart for deterministic and probabilistic scenarios, both of which contain forecast weather.

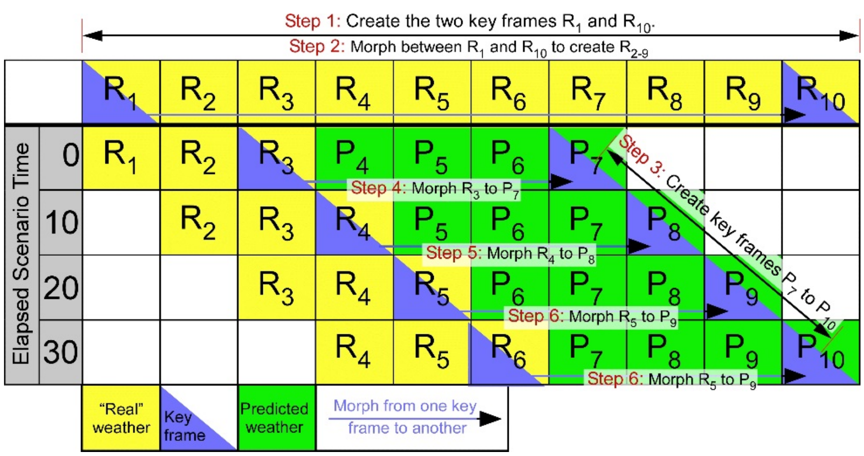


Figure 17. the sequencing chart for forecast weather

To generate scenarios with forecasts, we followed a basic algorithm. Figure 18 above and the list below illustrate:

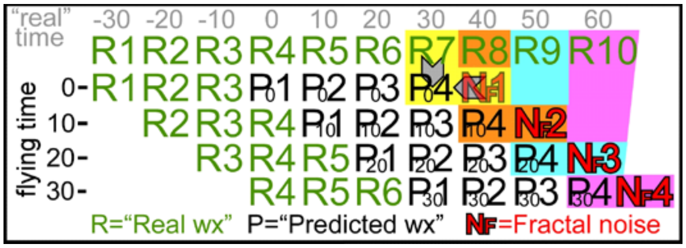


Figure 18. Another sequencing chart for forecast weather, showing how keyframes were morphed to create forecasts

1. Use the Weather Generator to create a starting-weather pattern for the keyframe R_1, and an ending-weather pattern for the keyframe R_{10}.
2. Create the remaining "R" frames by morphing smoothly from R_1 to R_{10} to generate the remaining eight intervening "R" historical frames.
3. Create the EST_0 sequence
    a. select $R_{1\text{-}3}$ as the three historical frames.
    b. set up R_3 as the left keyframe and R_7 as the right keyframe.
    c. add 20% fractal noise (explained below) to R_7 , perturbing it to represent prediction error. This creates prediction keyframe P_7.
    d. morph smoothly from R_3 to P_7 to generate the remaining prediction frames $P_{4\text{-}6}$. This creates P frames having prediction error increasing over time.
4. Create the $EST_{10\text{-}30}$ sequences by the same method as Step 3.

Fractal noise. The fractal noise used was quite simple, merely being another similar-looking frame created by the Weather Generator, using a different pseudorandom-number seed.

Deterministic Weather

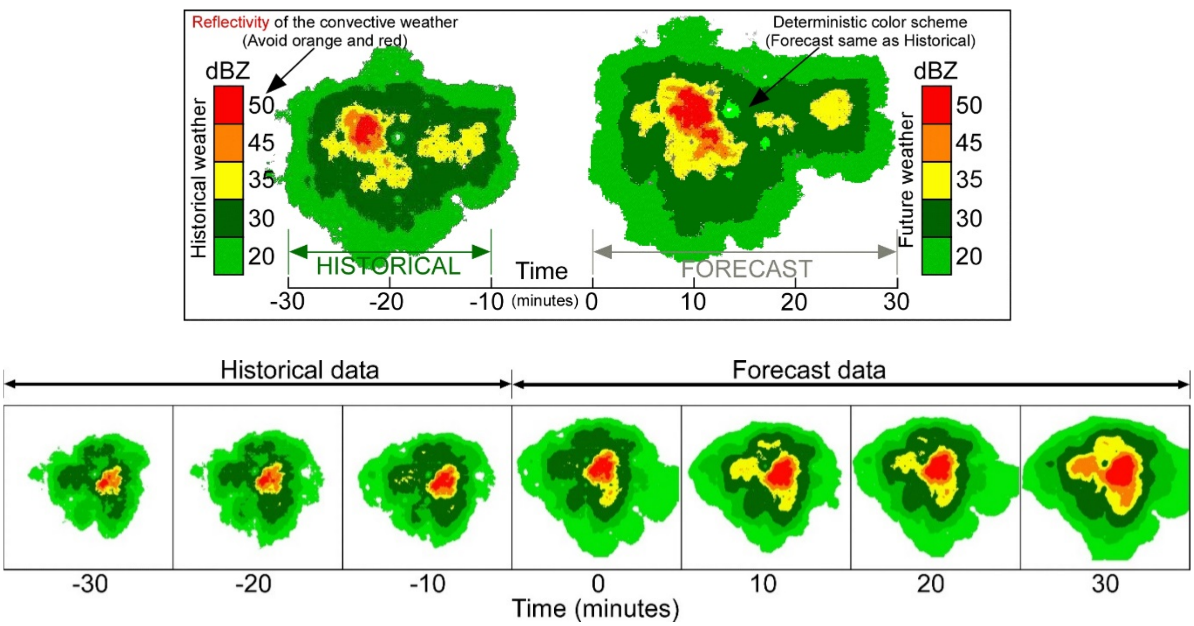
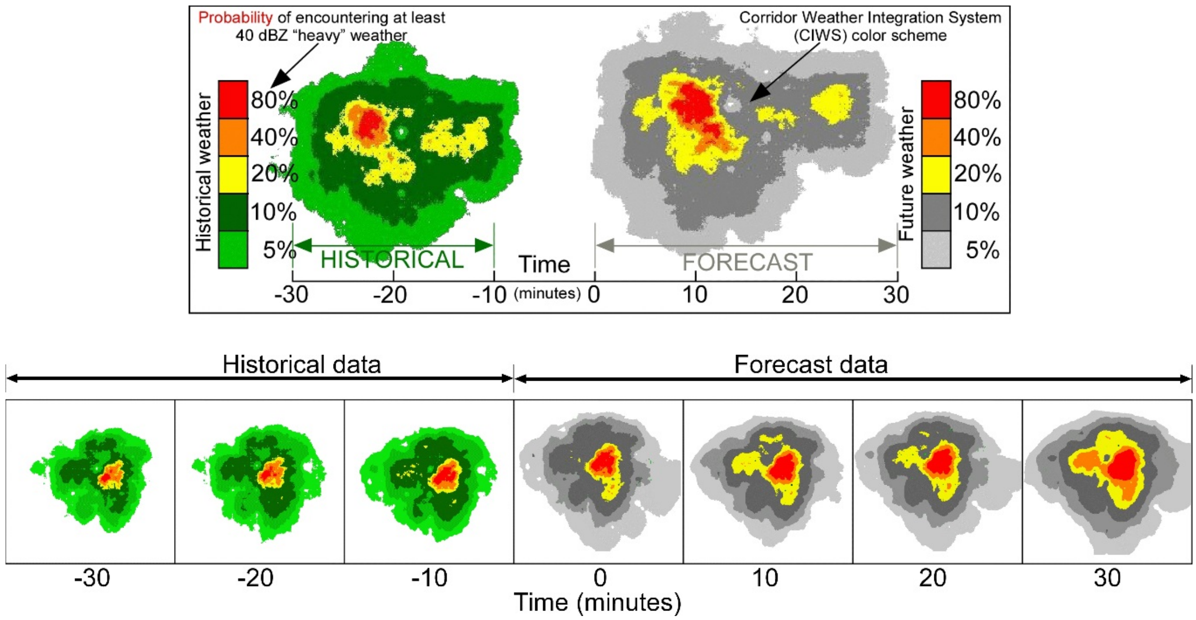


Figure 19.

Probabilistic Weather



a b c

Figure 20.

The Weather-Control Panel

Figure 20 shows a roughly half-size screenshot of the Weather Control Panel. This code allowed control over the size, pixel count for each color of both keyframes in a series, and rotation of the entire image. Complete *Mathematica* code is presented in Appendix B.

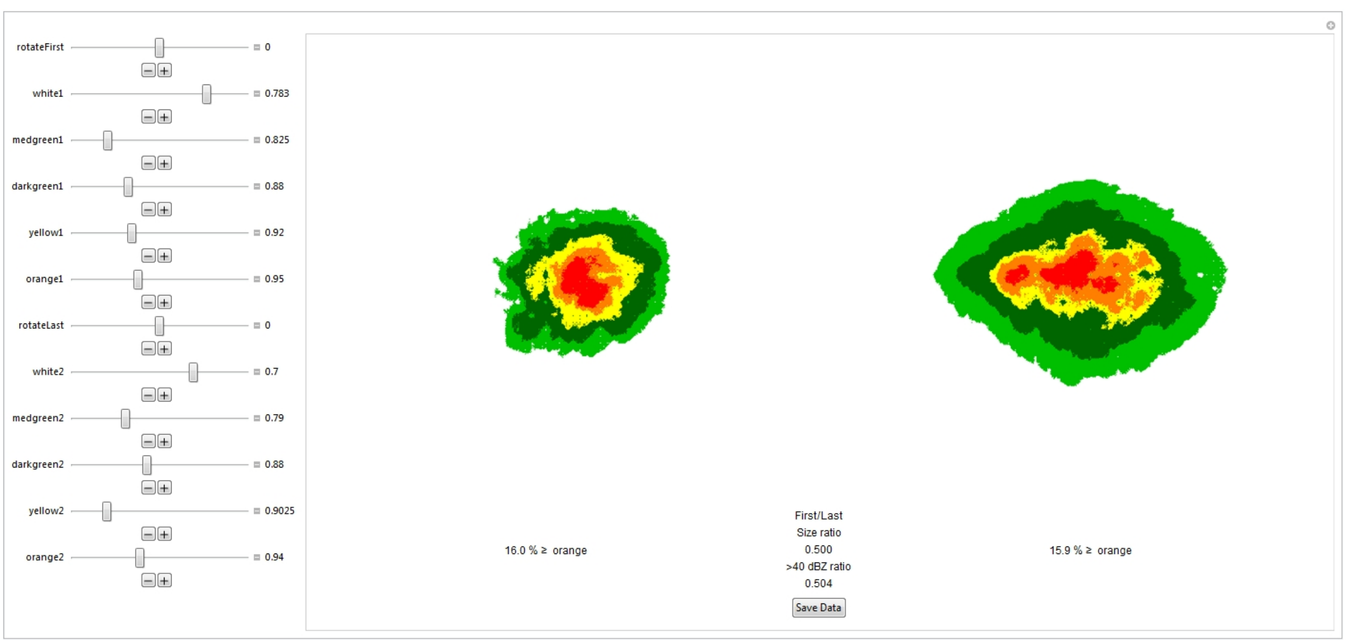


Figure 20. Weather Control Panel (45/100-scale), showing sliders that adjust color and rotation of beginning and ending keyframes.

The Weather Control Panel was deemed necessary because the primary weather-generation code allowed control over mainly the overall size of the array (513×513 pixels in this experiment) and the pseudorandom number seed, plus a few other ancillary parameters. But, it was also important to be able to control the *total number of non-white pixels* (i.e., the overall storm-cell size) and the *total number of heavy weather (orange and red) pixels*, as well as the *ratios* of overall size and heavy weather between keyframes.

Overall size and heavy weather size were considered vital in order to begin exploring which features of a storm constitute stimuli to pilots. Hypothetically, we want to be able to address questions like “Do what degree do pilots react to simply the overall size of a storm cell, as opposed to the size of the heavy weather within it?” and, “How do pilots react when the overall size of the storm cell changes, but its internal heavy weather doesn’t?”

APPENDIX B

Weather-Generation and Control Panel *Mathematica* Code

```
(* Step 1--First,  you need to have the properly-named folder to hold the data & images. Then, set the file path. Next, create the 10 "Real Weather" images. Create two keyframes, then
morph between them
If skipping Step 3, these still must be initialized.  If the RealWx[[ ]].jpg and .dat files already exist, can proceed to Step 4. If noise function NF[[ ]] frames exist, may proceed to Step 10.
If all RealWx and P4
frames are finished, you only need Step 12 a,b. *)
directoryName="H-LA";
CIWS=False;  (* Whether or not to draw prediction frames using the CIWS color scheme, where greens are replaced by grays *)
where="work";
If[ where==   "home",SetDirectory["C:\\Billy Bob\\Telework\\PROBWX\\"<>directoryName], SetDirectory[ "C:\\Billy Bob\\Projects Too Big for M Drive\\PROBWX\\"<>directoryName]]
FileNames[]
RealWx=Table[0,10];
x=9;  (* Number of recursion levels. Controls weather-image array size, which will be 2^x+1 *)
size=2^x+1;
n=0.65;  (* 0.55+RandomVariate[NormalDistribution[0,.05]]; Starting noise value *)
rM=0.56; (* 0.65  Range modifier for random number generator. Lower values make bigger, smoother, blobbier images *)
b=14.3;  (* 5.3 Floor value of initial RNG values for the first 4 corners. *)
wr=9.5/size; (* 5.5 Slope of the squashing function sq[ ]. The greater the numerator, the more squashing towards zero at the ends of the array *)
wc=6.5/size;  (* Squashes L-R *)

(*Step 2a--If you've already generated the two end-keyframes, load them here, and skip Step 3 below. Otherwise, go on to the Step 3 below, or to Step 10, if the NF files already exist
*)
RealWx[[1]]=Get["RealWx[[1]].dat"];
RealWx[[10]]=Get["RealWx[[10]].dat"];
(*Step 2b--If you already have all 10 RealWx images on file, and you only want to create a historical-wx "H-xx" set of "H-xx.jpgs", run this cell and you're done *)
For[i=1,i<=10,i++,RealWx[[i]]=Import[ StringJoin["RealWx[[",ToString[i],"]].jpg"]]];
imagesFinal=Table[0,{r,1,4},{c,1,3}];
For[r=1,r<=4,r++,
  For[c=1,c<=3,c++,
   imagesFinal[[r,c]]=RealWx[[r+(c-1)]];
   ]];
imagesFinal
AbsoluteTiming[Put[ imagesFinal, StringJoin[directoryName,".jpgs"]]]

(* Step 3--Otherwise, generate the two RealWx end frames (keyframes) here *)
(*   n=.0.65;      rM= 0.56;      b=14.3;      wr=9.5/size;      wc=6.5/size;  Below are the first / last keyframe seeds for RealWx images;
H-SM: 8 / 3;     H-ST: 9 / 13;      H-LA: 6 / 10;  Historical wx has only 3 frames per loop; Wx either gets SMaller, remains STatic, or gets LArger;
D-SM: 1 / 7;     D-ST: 24 / 23;      D-LA: 3 / 24;  Deterministic has 7 frames per loop;
P-SM: 5 / 13;     P-ST: 22 / 20;      P-LA: 3 / 2;  Probabilistic has 7 frames per loop;
SM=getting smaller over time;  ST=size is static over time;  LA=growing larger over time *)
ClearAll[g,sq,dataStructureName];
frameNum=1;  (* The number of the end keyframe you're about to generate *)
seed=5  ;    SeedRandom[seed];  (* Initialize the random-number generator with the specified seed *)
(* whichCF =myCFD  myCFD is the deterministic ColorFunction.  myCFP is the probabilistic *)
dataName=StringJoin["RP0s[[",ToString[frameNum],"]].dat"]         (* Use RealWx[[1]] or RealWx[[10]] *)
sq[r_,c_]:= N[4./((1+2^(wr Abs[r-size/2])) (1+2^(wc Abs[c-size/2]) ) ) ];(* A 2D squashing function that peaks □1.0 at the center of the 2D array of 0-1 values  *)
```

```
data=Table[0,{r,1,size},{c,1,size}];  (* Position {1,1} in ArrayPlot will be upper lefthand corner *)
data[[1,1]]=              sq[1,1]*RandomReal[{0,b}];
data[[1,size]]=       sq[1,size]*RandomReal[{0,b}];
data[[size,size]]= sq[size,size]*RandomReal[{0,b}];
data[[size,1]]=       sq[size,1]*RandomReal[{0,b}];

whichCF=Function[z,
  If[z<0.70,RGBColor[1,1,1],         (* white *)
   If[z<0.79,RGBColor[0,0.75,0],   (* med. green *)
    If[z<0.88,RGBColor[0,0.4,0],    (* dark green *)
     If[z<0.92,RGBColor[1,1,0],      (* yellow *)
      If[z<0.95,RGBColor[1,0.5,0],  (* orange, the color pilots must avoid *)
       RGBColor[1,0,0] ]]]]]] ;           (* red *)

g[r_,c_,0]:=0;     counter=0;
g[row_,col_,xlevel_]:=(*g[row,col,xlevel]=*)Module[{incr,siz,r,c,RNGrng},
  siz=2^xlevel+1;   incr=(siz-1)/2;    RNGrng=n rM^(x-xlevel);  (* <- range of the random number generator *)
  data[[row+incr,col+incr ]]=sq[row+incr,col+incr]*RandomReal[{0,RNGrng}]+(data[[row,col]]+data[[row,col+siz-1 ]]+data[[row+siz-1, col]]+data[[row+siz-1, col+siz-1 ]])/4; (*Midpt of ea
square *)

  If[data[[row,col+incr ]]== 0,data[[row,col+incr ]]=sq[row,col+incr]*RandomReal[{0,RNGrng}]+(data[[row,col]]+data[[row,col+siz-1 ]]+data[[row+incr,col+incr ]])/3]; (*The row midpoint, at
12 o'clock *)

  If[data[[row+siz-1,col+incr ]]==0,data[[row+siz-1,col+incr ]]=sq[row+siz-1,col+incr]*RandomReal[{0,RNGrng}]+(data[[row+siz-1,col]]+data[[row+siz-1, col+siz-1
]]+data[[row+incr,col+incr ]])/3];    (*The row midpoint, at 6 o'clock *)

  If[data[[row+incr,col+siz-1 ]]==0,data[[row+incr,col+siz-1 ]]=sq[row+incr,col+siz-1]*RandomReal[{0,RNGrng}]+(data[[row,col+siz-1]]+data[[row+siz-1,col+siz-
1]]+data[[row+incr,col+incr ]])/3];  (*The col midpoint, at 3 o'clock *)

  If[data[[row+incr,col ]]==0,data[[row+incr,col ]]=sq[row+incr,col]*RandomReal[{0,RNGrng}]+ (data[[row, col ]]+data[[row+siz-1, col ]]+data[[row+incr,col+incr ]])/3];  (*The col midpoint,
at 9 o'clock *)
  If[xlevel>1,
   For[r=row,r<= row+incr,r+=incr,
    For[c=col,c<=  col+incr,c+=incr,
     counter++;
     g[r,c,xlevel-1]
     ] ]
   ]
  ];  (* End declaration of g[ ]  *)
g[1,1,x];

dataStructureName=Table[0,{5}];
dataStructureName[[1]]={{Length[data],Length[data[[1]]]},{Ceiling[(Length[data])/2],Ceiling[(Length[data[[1]]])/2]}};
dataStructureName[[2]]= whichCF;
dataStructureName[[3]]={"name=",dataName,"  size=",size, "n=",n,"  rM=",rM, "  b=",b,"  wr=",wr," wc=",wc, "  seed=",seed};
dataStructureName[[4]]=0;  (* This will hold the value of "rotation", i.e., how many degrees the image will be rotated *)
dataStructureName[[5]]=data;
(* RECEIVING FILE --> *)  RealWx[[frameNum]]=dataStructureName;    (* Use RealWx[[1]],     RealWx[[10]],      RP0s[[7]],   RP10s[[7]],   RP20s[[7]],   or RP30s[[7]] *)
(*Print["Array size: ",size,", N function calls: ",counter,"   Generation time: ",t[[1]]];*)
img=ArrayPlot[dataStructureName[[5]],ColorFunction->whichCF,ColorFunctionScaling->True,Frame->False(*,Mesh→All*),  Axes->False,(*ImageSize→{Scaled[.5]},*) ImageSize->{2^x+1
```

```
,2^x+1 },AspectRatio->1/1]
RP0s[[1]].dat
(* Step 4--Now, test whether your two key endpoint frames--RealWx[[1,5]] and [[10,5]]--are functional *)
d1=ArrayPlot[RealWx[[1,5]],ColorFunction->RealWx[[1,2]],ColorFunctionScaling->True,Frame->False,ImageSize-> {2^x+1 ,2^x+1 } ];
d2=ArrayPlot[RealWx[[10,5]],ColorFunction->RealWx[[10,2]],ColorFunctionScaling->True,Frame->False,ImageSize-> {2^x+1 ,2^x+1 } ];
Row[{d1,d2}]
(* Step 5--Now, normalize the data as what ColorFunctionScaling[ ] does--make the minimum array value 0 and the maximum value 1  *)
dataFirst=RealWx[[1]]; dataLast=RealWx[[10]];          (* And, load the big-array *)
dataFirstMin=Min[dataFirst[[5]]];  dataFirstMax=Max[dataFirst[[5]]];   (* Rescale[ ] might also work here *)
dataFirst[[5]]=(dataFirst[[5]]-dataFirstMin)/(dataFirstMax-dataFirstMin);
dataLastMin=Min[dataLast[[5]]];  dataLastMax=Max[dataLast[[5]]];
dataLast[[5]]=(dataLast[[5]]-dataLastMin)/(dataLastMax-dataLastMin);
Print[Min[dataFirst[[5]]],"\t",Max[dataFirst[[5]]]"\n",Min[dataLast[[5]]],"\t",Max[dataLast[[5]]] ];  (* Check that the operation's been properly done *)

(* Step 6--Refine the endframes to spec. If this has already been done & saved to file, may proceed to Step 9.
All this stuff is a workaround to the fact that MMCA won't let us pass pointers *)
Clear[rotateFirst,white1,white2,medgreen1,medgreen2,rotateLast,darkgreen1,darkgreen2,orange1,orange2,yellow1,yellow2,orange1,orange2];
saveData[CF1_,CF2_]:=Module[{},
   dataFirst[[2]]=CF1;
   dataLast[[2]]=CF2;
   Print["ColorFunctions saved for dataFirst and dataLast"];
   ];
rotateFirst=0; rotateLast=0;
white1=w1Init=dataFirst[[2,2,1,2]];                white2=w2Init=dataLast[[2,2,1,2]];
medgreen1=mg1Init=dataFirst[[2,2,3,1,2]];    medgreen2=mg2Init=dataLast[[2,2,3,1,2]];
darkgreen1=dg1Init=dataFirst[[2,2,3,3,1,2]];  darkgreen2=dg2Init=dataLast[[2,2,3,3,1,2]];
yellow1=y1Init=dataFirst[[2,2,3,3,3,1,2]];       yellow2=y2Init=dataLast[[2,2,3,3,3,1,2]];
orange1=o1Init=dataFirst[[2,2,3,3,3,3,1,2]];   orange2=o2Init=dataLast[[2,2,3,3,3,3,1,2]];
myElements={"StepLeftButton","StepRightButton","InlineInputField"};  (* "SnapshotButton" ? *)
myTrackedSymbols={rotateFirst,white1,medgreen1,darkgreen1,yellow1,orange1,rotateLast,white2,medgreen2,darkgreen2,yellow2,orange2};

Manipulate[  (* Now, we are going to manipulate the ColorFunction to refine our keyframes *)
              (* Below, we have to write the color function using placeholders, because the previous method proved "impermeable" to the actual values of colors in RAM during write-to-
file *)
 whichCF1=Function[z,
     If[z<#1,RGBColor[1,1,1],       (*white*)
      If[z<#2,RGBColor[0,0.75,0]  (*medium green*)
       If[z<#3, RGBColor[0,0.4,0], (*dark green*)
        If[z<#4,RGBColor[1,1,0],    (*yellow*)
         If[z<#5,RGBColor[1,0.5,0],  (* orange, the color pilots must avoid *)
          RGBColor[1,0,0] ]]]]]]&[white1,medgreen1,darkgreen1,yellow1,orange1] ;    (* red *)
 whichCF2=Function[z,
     If[z<#1,RGBColor[1,1,1],
      If[z<#2,RGBColor[0,0.75,0],
       If[z<#3, RGBColor[0,0.4,0],
        If[z<#4,RGBColor[1,1,0],
         If[z<#5,RGBColor[1,0.5,0],  (* orange, the color pilots must avoid *)
          RGBColor[1,0,0] ]]]]]]&[white2,medgreen2,darkgreen2,yellow2,orange2] ;    (* red *)
 counterFirstOrangeRed=counterLastOrangeRed=counterFirstNTot=counterLastNTot=0; (* Used to count pixels of hazard areas *)
```

```
For[r=1,r< size,r++,  (* Count the total # of colored pixels, and the # of orange+red pixels *)
 For[c=1,c< size,c++,
  tempFirst=dataFirst[[5,r,c]];  tempLast=dataLast[[5,r,c]];
  If[ tempFirst>=yellow1,counterFirstOrangeRed++; counterFirstNTot++, If[ tempFirst>=white1,counterFirstNTot++]];
  If[ tempLast>=yellow2,counterLastOrangeRed++; counterLastNTot++, If[ tempLast>=white2,counterLastNTot++]];
  ]];
pCentFirst=Graphics[{Text[Style[StringJoin[ToString[N[100*counterFirstOrangeRed/counterFirstNTot,3]]," % >=  orange"],Medium]]},ImageSize->{120,20}];
pCentLast=Graphics[{Text[Style[StringJoin[ToString[N[100*counterLastOrangeRed/counterLastNTot,3]]," % >=  orange"],Medium]]},ImageSize->{120,20}];
sizeRatio=N[counterFirstNTot/counterLastNTot,3];
dBZRatio=N[counterFirstOrangeRed/counterLastOrangeRed,3];
pixelRatios=Graphics[{  Text[  Style[StringJoin["First/Last\nSize ratio\n", ToString[sizeRatio],"\n>40 dBZ ratio\n",ToString[dBZRatio] ],   Medium  ]  ]},  ImageSize->{80,95} ];

g1=Rotate[ArrayPlot[dataFirst[[5]],ColorFunction->whichCF1,ColorFunctionScaling->False,Frame->False,ImageSize->{2^x+1 ,2^x+1 } ], rotateFirst Degree];
g2=Rotate[ArrayPlot[dataLast[[5]],ColorFunction->whichCF2,ColorFunctionScaling->False,Frame->False,ImageSize->{2^x+1 ,2^x+1} ], rotateLast Degree];
Grid[{{g1,Spacer[10],g2},
  {pCentFirst,pixelRatios,pCentLast},
  {Spacer[10],Button["Save Data",saveData[whichCF1,whichCF2] ,ImageSize->Medium  ]  }
  }],
{{rotateFirst,0,"rotateFirst"},-90,90,Appearance->"Open",AppearanceElements->myElements},
{{white1,w1Init,"white1"},0,1,0.005,Appearance->"Open",AppearanceElements->myElements},
{{medgreen1,mg1Init,"medgreen1"},white1, 1,0.005, Appearance->"Open",AppearanceElements->myElements},
{{darkgreen1,dg1Init, "darkgreen1"},medgreen1,1,0.005,Appearance->"Open",AppearanceElements->myElements},
{{yellow1,y1Init,"yellow1"},darkgreen1,1,0.0025,Appearance->"Open",AppearanceElements->myElements},
{{orange1,o1Init,"orange1"},yellow1,1,0.0025,Appearance->"Open",AppearanceElements->myElements},
{{rotateLast,0,"rotateLast"},-90,90,1,Appearance->"Open",AppearanceElements->myElements},
{{white2,w2Init,"white2"},0,1,0.005,Appearance->"Open",AppearanceElements->myElements},
{{medgreen2,mg2Init,"medgreen2"},white2,1,0.005,Appearance->"Open",AppearanceElements->myElements},
{{darkgreen2,dg2Init,"darkgreen2"},medgreen2,1,0.005,Appearance->"Open",AppearanceElements->myElements},
{{yellow2,y2Init,"yellow2"},darkgreen2,1,0.0025,Appearance->"Open",AppearanceElements->myElements},
{{orange2,o2Init,"orange2"},yellow2,1,0.0025,Appearance->"Open",AppearanceElements->myElements},
TrackedSymbols:>myTrackedSymbols,ControlPlacement->Left,ContinuousAction->False, LocalizeVariables->False  (* ←Have to let Manipulate use global variables *)
]
(* Step 7--Now, use the keyframes to create the remaining RealWx images, morphing from the first keyframe to the last *)
(*The proportions we want are    i = 1   2  3   4  5   6  7   8  9  10
                                 dataFirst   9/9  8/9  7/9  6/9  5/9  4/9  3/9  2/9  1/9  0/9
                                 dataLast   0/9  1/9  2/9  3/9  4/9  5/9  6/9  7/9  8/9  9/9  *)
(*  Be sure that dataFirst and dataLast are meaningful, REALWX DATA SHOULD REMAIN UNCROPPED *)
firstKeyCell=1; (* number of the 1st  keycell. NOTE:  The first and last keycells will remain unmodified. Intervening cells will be morphed *)
lastKeyCell=10;  (* number of the last keycell in your dataset, e.g., RealWx □10 *)
images=dataMorphed=rotationMorphed=Table[0,lastKeyCell];
temp={0,0,0,0,0};
temp3=lastKeyCell-firstKeyCell;
For[  i=firstKeyCell,i<=  lastKeyCell,i++,  (* Morph the first image into the last *)
                                 temp1=lastKeyCell-i;    temp2=i-firstKeyCell;   (* <-- should be temp=lastKeyCell+1-i;   temp2=i-firstKeyCell; *)
  rotationMorphed[[i]]= (  temp1*dataFirst[[4]] +     temp2*dataLast[[4]]  )/  temp3;  (* Currently, rotation isn't fully implemented *)
  dataMorphed[[i]]=        (  temp1*dataFirst[[5]] +  temp2*dataLast[[5]]  )/temp3;
  white3=                          (  temp1*white1 +  temp2*white2 )/temp3;
  medgreen3=                     (  temp1*medgreen1 + temp2*medgreen2 )/temp3;
  darkgreen3=                    (  temp1*darkgreen1 + temp2*darkgreen2 )/temp3;
  yellow3=                          (  temp1*yellow1 +   temp2*yellow2 )/temp3;
```

```
  orange3=                          (  temp1*orange1 +   temp2*orange2 )/temp3;

  whichCF3=Function[z,
     If[z<#1,RGBColor[1,1,1],
      If[z<#2,RGBColor[0,0.75,0],
       If[z<#3, RGBColor[0,0.4,0],
        If[z<#4,RGBColor[1,1,0],
         If[z<#5,RGBColor[1,0.5,0],  (* orange, the color pilots must avoid *)
          RGBColor[1,0,0] ]]]]]]&[white3,medgreen3,darkgreen3,yellow3,orange3] ;

  img=ArrayPlot[dataMorphed[[i]],ColorFunction->whichCF3,ColorFunctionScaling->False,Frame->False,  Axes->False ];
  (*img=ImageCrop[img];*)
  imgXY=ImageDimensions[img];
  temp[[1]] ={imgXY,{Ceiling[imgXY[[1]]/2],Ceiling[imgXY[[2]]/2]}}; (* Image dimensions and midpoint *)
  temp[[2]] =whichCF3;
  If[i==firstKeyCell,RealWx[[i,3]]=dataFirst[[3]]];    If[i==lastKeyCell,RealWx[[i,3]]=dataLast[[3]]];
  temp[[4]] =rotationMorphed[[i]];
  temp[[5]] =dataMorphed[[i]];
  RealWx[[i]]=temp;
  images[[i]]=Rotate[img, rotationMorphed[[i]] Degree];  (* images[[ i ]] is still 513x513 *)
  ];  (* Now, check the images, to make sure they came out all right before saving them *)
GraphicsGrid[ { {images[[1]],   images[[2]],   images[[3]],   images[[4]] ,images[[5]]},  { images[[6]], images[[7]],images[[8]], images[[9]],images[[10]] }},   ImageSize->{1000,400} ]

(* Step 8--Save the images and data to file *)

saveData[outFile_,theData_]:=Module[{},
   ptr=OpenWrite[outFile];
   Write[ptr,theData];
   Close[outFile];
   ];

PrintTemporary[ProgressIndicator[Dynamic[i],{firstKeyCell,lastKeyCell}]];
For[  i=firstKeyCell,i<=  lastKeyCell,i++,
  saveData[ StringJoin[ "RealWx[[",ToString[i],"]].dat"] ,RealWx[[i]] ];      (* Could also have used Put[ ] *)
  Export[StringJoin[ "RealWx[[",ToString[i],"]].jpg"],images[[i]]]  (* BE ADVISED THAT EXPORT[ ] AUTO-CROPS JPGs *)
  ];
 CloudObject`Private`i
(* Step 9--Generate 4 "Prediction error" endframes (fractal noise keyframes), 1 each for t=0,10,20,30. These will be mixed in w the RealWx, to represent prediction error increasing w
time.
These will be named NF[[1]] ... NF[[4]].dat and .jpg. IF these already exist, proceed to Step 10a to load them *)
(* D-SM: {7,4,10,8}     D-ST: {10,7,6,4};      D-LA: {4,6,7,10};
P-SM: {25,9,12,15};     P-ST: {4,8,7,2};      P-LA: {2,4,8,20};   Random-number seed sets for the 4 error frames *)

seedSet={7,4,10,8};
numFrames=4; (* Number of NF error frames to generate *)
NFEndFrames=Table[0,numFrames];  (* To hold the t=0, 10, 20, 30 position-7 end-keyframes *)
For[errorFrameNum=1,errorFrameNum<=numFrames,errorFrameNum++,  (* Generate 4 error "noise" frames.  1 □NF1 ... 4 □NF4  *)
 SeedRandom[seedSet[[errorFrameNum]]]; (* Initialize the random number generator *)
 dataName=StringJoin["NFEndFrames[[",ToString[errorFrameNum],"]].dat"] ;
 sq[r_,c_]:= N[4./((1+2^(wr Abs[r-size/2])) (1+2^(wc Abs[c-size/2]) ) ) ];(* A 2D squashing function that peaks □1.0 at the center of the 2D array of 0-1 values  *)
```

```
data=Table[0,{r,1,size},{c,1,size}];  (* Position {1,1} in ArrayPlot will be upper lefthand corner *)
data[[1,1]]=              sq[1,1]*RandomReal[{0,b}];
data[[1,size]]=       sq[1,size]*RandomReal[{0,b}];
data[[size,size]]= sq[size,size]*RandomReal[{0,b}];
data[[size,1]]=       sq[size,1]*RandomReal[{0,b}];

whichCF=Function[z,
  If[z<0.70,RGBColor[1,1,1],        (* white *)
   If[z<0.79,RGBColor[0,0.75,0],   (* med. green *)
    If[z<0.88,RGBColor[0,0.4,0],    (* dark green *)
     If[z<0.92,RGBColor[1,1,0],      (* yellow *)
      If[z<0.95,RGBColor[1,0.5,0],  (* orange, the color pilots must avoid *)
       RGBColor[1,0,0] ]]]]] ;           (* red *)

g[r_,c_,0]:=0;     counter=0;
g[row_,col_,xlevel_]:=(*g[row,col,xlevel]=*)Module[{incr,siz,r,c,RNGrng},
  siz=2^xlevel+1;   incr=(siz-1)/2;    RNGrng=n rM^(x-xlevel);  (* <- range of the random number generator *)
  data[[row+incr,col+incr ]]=sq[row+incr,col+incr]*RandomReal[{0,RNGrng}]+(data[[row,col]]+data[[row,col+siz-1 ]]+data[[row+siz-1, col]]+data[[row+siz-1, col+siz-1 ]])/4; (*Midpt of ea square *)

  If[data[[row,col+incr ]]== 0,data[[row,col+incr ]]=sq[row,col+incr]*RandomReal[{0,RNGrng}]+(data[[row,col]]+data[[row,col+siz-1 ]]+data[[row+incr,col+incr ]])/3]; (*The row midpoint, at 12 o'clock *)

  If[data[[row+siz-1,col+incr ]]==0,data[[row+siz-1,col+incr ]]=sq[row+siz-1,col+incr]*RandomReal[{0,RNGrng}]+(data[[row+siz-1,col]]+data[[row+siz-1, col+siz-1 ]]+data[[row+incr,col+incr ]])/3];    (*The row midpoint, at 6 o'clock *)

  If[data[[row+incr,col+siz-1 ]]==0,data[[row+incr,col+siz-1 ]]=sq[row+incr,col+siz-1]*RandomReal[{0,RNGrng}]+(data[[row,col+siz-1]]+data[[row+siz-1,col+siz-1]]+data[[row+incr,col+incr ]])/3];  (*The col midpoint, at 3 o'clock *)

  If[data[[row+incr,col ]]==0,data[[row+incr,col ]]=sq[row+incr,col]*RandomReal[{0,RNGrng}]+ (data[[row, col ]]+data[[row+siz-1, col ]]+data[[row+incr,col+incr ]])/3];  (*The col midpoint, at 9 o'clock *)
  If[xlevel>1,
   For[r=row,r<= row+incr,r+=incr,
    For[c=col,c<=  col+incr,c+=incr,
     counter++;
     g[r,c,xlevel-1]
     ] ]
   ]
  ];  (* End declaration of g[ ]  *)
g[1,1,x];

     dataStructureName=Table[0,{5}];
dataStructureName[[1]]={{Length[data],Length[data[[1]]]},{Ceiling[(Length[data])/2],Ceiling[(Length[data[[1]]])/2]}};
dataStructureName[[2]]= whichCF;
      dataStructureName[[3]]={"name=",dataName,"  size=",size, "n=",n,"  rM=",rM, "  b=",b,"  wr=",wr," wc=",wc, "  seed=",seed};
dataStructureName[[4]]=0;  (* This will hold the value of "rotation", i.e., how many degrees the image will be rotated *)
dataStructureName[[5]]=Rescale[data];  (* Normalize data so that Min[data]=0 and Max[data]=1 *)
(* RECEIVING FILE --> *)  NFEndFrames[[errorFrameNum]]=dataStructureName;
img=ArrayPlot[dataStructureName[[5]],ColorFunction->whichCF,ColorFunctionScaling->False,Frame->False,  Axes->False,ImageSize->{2^x+1 ,2^x+1 },AspectRatio->1/1];
Print[ToString[errorFrameNum],"\t",ToString[seedSet[[errorFrameNum]]],img];
```

```
 fNameBase=StringJoin[ "NF[[",ToString[errorFrameNum],"]]"];
 Put[dataStructureName,StringJoin[ fNameBase,".dat"]];
 Export[StringJoin[ fNameBase,".jpg"],img]  (* BE ADVISED THAT EXPORT[ ] AUTO-CROPS IMAGES--except that this didn't?? <- CHECK *)
 ];  (* End    outer For[errorFrameNum=1,  loop *)

(* Step 10a--If NF error frames already exist, Get[ ] them here. If RealWx, NF and P4 frames all exist, go to Step 12a to load them.
 IF skipping straight to here from an earlier step, ensure that RealWx (from Step 2), plus numframes, whichCF,
and NFEndFrames (all from Step 9) have been defined by executing Step 10a here. Otherwise, SKIP THIS STEP and proceed to Step 10b*)
numFrames=4;
whichCF=Function[z,
   If[z<0.70,RGBColor[1,1,1],        (* white *)
    If[z<0.79,RGBColor[0,0.75,0],   (* med. green *)
     If[z<0.88,RGBColor[0,0.4,0],    (* dark green *)
      If[z<0.92,RGBColor[1,1,0],      (* yellow *)
       If[z<0.95,RGBColor[1,0.5,0],  (* orange, the color pilots must avoid *)
        RGBColor[1,0,0] ]]]]] ;          (* red *)
NFEndFrames=Table[0,numFrames];  (* To hold the t=0, 10, 20, 30 position-7 end-keyframes *)
PrintTemporary[ProgressIndicator[Dynamic[errorFrameNum],{1,numFrames}]];
For[errorFrameNum=1,errorFrameNum<=numFrames,errorFrameNum++,
  NFEndFrames[[errorFrameNum]]=Get[StringJoin["NF[[",ToString[errorFrameNum],"]].dat"  ]]
  ];
PrintTemporary[ProgressIndicator[Dynamic[i],{1,10}]];
For[i=1,i<=10,i++,RealWx[[i]]=Get[ StringJoin["RealWx[[",ToString[i],"]].dat"]]]; (*  EXPORT[ ] SOMETIMES AUTO-CROPS JPGs *)
(* Step 10b--Blend the NF error endframes with the 4 RealWx frames to create the P4 error endframes. E.g., R7 ( RealWx[[7,5]] ) blends w N1 ( NF[[1]] ) to create P4 for t = 0 (P0[[4]] )
*)
(*Do NOT run 10c if you've just Get [] pre-existing P4s, or you'll erase the P4Frames array--instead, go to 10d *)
P4Frames=Table[0,{i,numFrames},{j,5}];  (* To hold the blended (NF + R) end-keyframes *)
tempImages=Table[0,{i,3},{j,numFrames}];
blendProp=0.2;  (* The proportion of fractal noise to add to each RealWx frame to generate the P4 frame *)
For[errorFrameNum=1,errorFrameNum<=numFrames,errorFrameNum++,  (* Generate 4 error "noise" frames.  1 □NF1 ... 4 □NF4  *)
  (* fNameBase=StringJoin["P",ToString[10 (errorFrameNum-1)],"[[4]]"] ; *)

  For[j=1,j<=4,j++,P4Frames[[errorFrameNum,j]] =RealWx[[errorFrameNum+6,j ]]  ];  (* Fill in the other 4 data fields, based on what's in RealWx *)
              (* tempImages[[1]] is the RealWx image. tempImages[[2]] is the NF fractal noise.  tempImages[[3]] is the P4 frame, which is the sum of RealWx + NF *)
  tempImages[[1,errorFrameNum]]=ArrayPlot[RealWx[[errorFrameNum,5]],ColorFunction->RealWx[[errorFrameNum,2]],ColorFunctionScaling->False,Frame->False,  Axes->False ];
  tempImages[[2,errorFrameNum]]=ArrayPlot[NFEndFrames[[errorFrameNum,5]],ColorFunction->NFEndFrames[ [errorFrameNum,2]]  ,ColorFunctionScaling->False,Frame->False,
Axes->False ];(*tempImages[[2,errorFrameNum]]=ArrayPlot[NFEndFrames[[errorFrameNum,5]],ColorFunction□whichCF (*RealWx[[errorFrameNum,2]] *)
,ColorFunctionScaling□True,Frame□False,  Axes□False ];*)
     P4Frames[[errorFrameNum,5]]=(1-blendProp)*RealWx[[errorFrameNum+2,5]] + blendProp*NFEndFrames[[errorFrameNum,5]]; (* Blend the real data and the error data *)
      P4Frames[[errorFrameNum,5]]=Rescale[P4Frames[[errorFrameNum,5]]];  (* Renormalize the data array to have Min=0, Max=1 *)
      (*Print[StringJoin["Min[P4Frames[[",ToString[errorFrameNum],"]]: "],Min[P4Frames[[errorFrameNum,5]]],  StringJoin["
Max[P4Frames[[",ToString[errorFrameNum],"]]"],Max[P4Frames[[errorFrameNum,5]]]]];*)
  tempImages[[3,errorFrameNum]]=ArrayPlot[P4Frames[[errorFrameNum,5]],ColorFunction->RealWx[[errorFrameNum+6,2]],ColorFunctionScaling->False,Frame->False,  Axes->False
]
  ];  (*  And, below, now visually check the results of adding noise N to RealWx signal R *)

GraphicsGrid[{{Graphics[Text[""]],Graphics[Text["P0[[4]]"]] ,Graphics[Text["P10[[4]]"]],Graphics[Text["P20[[4]]"]],Graphics[Text["P30[[4]]"]] },
  {Graphics[Text["RealWx (S)"]],tempImages[[1,1]],tempImages[[1,2]],tempImages[[1,3]],tempImages[[1,4]]},
        {Graphics[Text["Fractal noise (N)"]],tempImages[[2,1]],tempImages[[2,2]],tempImages[[2,3]],tempImages[[2,4]]} ,
  {Graphics[Text[StringJoin["Sum of ",ToString[1-blendProp],"S" + ",ToString[blendProp]," N "]]],tempImages[[3,1]],tempImages[[3,2]],tempImages[[3,3]],tempImages[[3,4]]}},ImageSize-
```

```
>{1200,750}]
(* Step 11--If RealWx + NF results look good, clean up the P4 endframes in another Control Panel to end up the desired overall size ratios and "≥ orange" ratios for all P4 endframes.
You will have to run this 4 times, one each for et=0 ( P4[[1]] ) ... to et=30 ( P4[[4]] ) *)
et = 00;  (* The elapsed flying time we're interested in tweaking the P4 S+N endframe, either t = 0, 10, 20, or 30 *)
button1Name=StringJoin["Update P4 CF, t = ",ToString[et]];
button2Name=StringJoin["Save P4.dat/jpg, t = ",ToString[et]];
P4datFilename=StringJoin[ "P4[[",ToString[IntegerPart[et/10]+1],"]].dat"];
P4jpgFilename=StringJoin[ "P4[[",ToString[IntegerPart[et/10]+1],"]].jpg"];
Clear[rotateLast,white2,medgreen2,darkgreen2,orange2,yellow2,orange2];
updateP4CF[CF2_]:=Module[{},  P4Frames[[IntegerPart[et/10]+1,2]]=CF2  ];
saveP4datjpg[CF2_]:=Module[{},
   Print["Saving"];
   updateP4CF[CF2];  (* Update the P4 data structure's ColorFunction *)
   Export[P4jpgFilename,g2];  (* BE ADVISED THAT EXPORT[ ] AUTO-CROPS IMAGES *)
   Put[ P4Frames[[IntegerPart[et/10]+1]], P4datFilename];
   Print["Done"]
   ];

dataFirst=RealWx[[IntegerPart[et/10]+3 ]];     dataLast=P4Frames[[IntegerPart[et/10]+1]];
rotateFirst=0; rotateLast=0;  (* Rotation isn't fully implemented yet *)
white1=dataFirst[[2,2,1,2]];                          white2=w2Init=dataLast[[2,2,1,2]];
medgreen1=dataFirst[[2,2,3,1,2]];                medgreen2=mg2Init=dataLast[[2,2,3,1,2]];
darkgreen1=dataFirst[[2,2,3,3,1,2]];             darkgreen2=dg2Init=dataLast[[2,2,3,3,1,2]];
yellow1=dataFirst[[2,2,3,3,3,1,2]];              yellow2=y2Init=dataLast[[2,2,3,3,3,1,2]];
orange1=dataFirst[[2,2,3,3,3,3,1,2]];            orange2=o2Init=dataLast[[2,2,3,3,3,3,1,2]];
myElements={"StepLeftButton","StepRightButton","InlineInputField"};  (* "SnapshotButton" ? *)
myTrackedSymbols={rotateLast,white2,medgreen2,darkgreen2,yellow2,orange2};
whichCF1=dataFirst[[2]];
counterFirstOrangeRed=counterFirstNTot=0; (* Used to count pixels of hazard areas *)
For[r=1,r< size,r++,  (* Count the total # of colored pixels, and the # of orange+red pixels *)
  For[c=1,c< size,c++,
   tempFirst=dataFirst[[5,r,c]];
   If[ tempFirst>=yellow1,counterFirstOrangeRed++; counterFirstNTot++, If[ tempFirst>=white1,counterFirstNTot++]]
   ]];

Manipulate[
 whichCF2=Function[z,
     If[z<#1,RGBColor[1,1,1],
      If[z<#2,If[CIWS==False,RGBColor[0,0.75,0],  GrayLevel[0.75]],
       If[z<#3,  If[CIWS==False,RGBColor[0,0.4,0],  GrayLevel[0.50]],
        If[z<#4,RGBColor[1,1,0],
         If[z<#5,RGBColor[1,0.5,0],  (* orange, the color pilots must avoid *)
          RGBColor[1,0,0] ]]]]]]&[white2,medgreen2,darkgreen2,yellow2,orange2] ;    (* red *)
 counterLastOrangeRed=counterLastNTot=0;

 For[r=1,r< size,r++,  (* Count the total # of colored pixels, and the # of orange+red pixels *)
  For[c=1,c< size,c++,
    tempLast=dataLast[[5,r,c]];
   If[ tempLast>=yellow2,counterLastOrangeRed++; counterLastNTot++, If[ tempLast>=white2,counterLastNTot++]]
   ]];
```

```
 pCentFirst=GraphicsColumn[{Graphics[{Text[Style[StringJoin["RealWx[[",ToString[IntegerPart[et/10]+3],"]]"],Medium]]},ImageSize->{120,20}],Graphics[{Text[Style[StringJoin[ToString[N[100*counterFirstOrangeRed/counterFirstNTot,3]]," % >=  orange"],Medium]]},ImageSize->{120,20}]
    }];
 pCentLast=GraphicsColumn[{Graphics[{Text[Style[StringJoin["P",ToString[IntegerPart[et]],"[[4]]"],Medium]]},ImageSize->{120,20}],
    Graphics[{Text[Style[StringJoin[ToString[N[100*counterLastOrangeRed/counterLastNTot,3]]," % >=  orange"],Medium]]},ImageSize->{120,20}]
    }];
 sizeRatio=N[counterFirstNTot/counterLastNTot,3];
 dBZRatio=N[counterFirstOrangeRed/counterLastOrangeRed,3];
 pixelRatios=Graphics[{  Text[  Style[StringJoin["First/Last\nSize ratio\n", ToString[sizeRatio],"\n>40 dBZ ratio\n",ToString[dBZRatio] ],   Medium  ]  ]},  ImageSize->{80,95} ];

 g1=Rotate[ArrayPlot[dataFirst[[5]],ColorFunction->whichCF1,ColorFunctionScaling->False,Frame->False,ImageSize->{size,size} ], rotateFirst Degree];  (* CHECK TO SEE IF ImageSize may have caused trouble in other Manipulates *)
 g2=Rotate[ArrayPlot[dataLast[[5]],ColorFunction->whichCF2,ColorFunctionScaling->False,Frame->False,ImageSize->{size,size} ], rotateLast Degree];
 Grid[{{g1,Spacer[10],g2},
   {pCentFirst,pixelRatios,pCentLast},
   {Spacer[10], Button[button1Name,updateP4CF[whichCF2] ,ImageSize->Medium  ],
    Button[button2Name, saveP4datjpg[whichCF2],ImageSize->Medium, Method->"Queued"]
    }(* For unknown reason, won't write complete file to disk w/o Method"Queued". Produces a very slow write, so be patient *)
   }],
 {{rotateLast,0,"rotateLast"},-90,90,1,Appearance->"Open",AppearanceElements->myElements},
 {{white2,w2Init,"white2"},0,1,0.005,Appearance->"Open",AppearanceElements->myElements},
 {{medgreen2,mg2Init,"medgreen2"},white2,1,0.005,Appearance->"Open",AppearanceElements->myElements},
 {{darkgreen2,dg2Init,"darkgreen2"},medgreen2,1,0.005,Appearance->"Open",AppearanceElements->myElements},
 {{yellow2,y2Init,"yellow2"},darkgreen2,1,0.001,Appearance->"Open",AppearanceElements->myElements},
 {{orange2,o2Init,"orange2"},yellow2,1,0.002,Appearance->"Open",AppearanceElements->myElements},
 TrackedSymbols:>myTrackedSymbols,ControlPlacement->Left,ContinuousAction->False, ContentSize->Automatic, LocalizeVariables->False  (* ←Have to let Manipulate use global variables *)
 ]
 Saving
 Done
(* Step 12a--If you already have the RealWx and P4 keyframes on-file, but not in RAM, load them here. Or, if everything's already in RAM, skip this step and go to Step 12b*)
RealWx=Table[0,10];
PrintTemporary[ProgressIndicator[Dynamic[i],{1,10}]];
For[i=1,i<=10,i++,RealWx[[i]]=Get[ StringJoin["RealWx[[",ToString[i], "]].dat"]]];
P4Frames=Table[0,4];
PrintTemporary[ProgressIndicator[Dynamic[j],{1,4}]];
For[j=1,j<=4,j++,P4Frames[[j]]=Get[ StringJoin["P4[[",ToString[j], "]].dat"]]];
(* Step 12b--Now, use RealWx and P4 keyframes to create the remaining row images, morphing from each first keyframe to the last *)
(*  Be sure that dataFirst and dataLast are meaningful, *)
(*The proportions we want are    R  P1 P2 P3 P4  but we already have the R and P4 frames
                                 so iterate p = 1   2    3
                                 dataFirst   4/4   3/4   2/4   1/4    0/4
                                 dataLast   0/4   1/4   2/4   3/4    4/4   *)
imagesFinal=Table[0,{r,1,4},{c,1,7}];
temp3=4;
PrintTemporary[ProgressIndicator[Dynamic[t],{0,3}]];
For[  t=0,t<=3,t++,  (* Morph the P elements corresponding to time = 0, 10, 20, 30 minutes*)
  dataFirst=RealWx[[t+3]];                    dataLast=P4Frames[[t+1]];  (* Image data for these should both be size x size *)
  white1=dataFirst[[2,2,1,2]] ;               white2=dataLast[[2,2,1,2]];
  medgreen1=dataFirst[[2,2,3,1,2]];      medgreen2=dataLast[[2,2,3,1,2]];
```

```
  darkgreen1=dataFirst[[2,2,3,3,1,2]];   darkgreen2=dataLast[[2,2,3,3,1,2]];
  yellow1=dataFirst[[2,2,3,3,3,1,2]];      yellow2=dataLast[[2,2,3,3,3,1,2]];
  orange1=dataFirst[[2,2,3,3,3,3,1,2]]; orange2=dataLast[[2,2,3,3,3,3,1,2]];
  (* NOTE: NEVER DID SOLVE THE PROBLEM OF EXPORT CROPPING *)
  rotationMorphed=0;
  imagesFinal[[t+1,1]]=ImageCrop[Rasterize[Rotate[ArrayPlot[RealWx[[t+3,5]],ColorFunction->RealWx[[t+3,2]],ColorFunctionScaling->False,Frame->False,  Axes->False
],rotationMorphed Degree]]];
  imagesFinal[[t+1,2]]=ImageCrop[Rasterize[Rotate[ArrayPlot[RealWx[[t+3,5]],ColorFunction->RealWx[[t+3,2]],ColorFunctionScaling->False,Frame->False,  Axes->False
],rotationMorphed Degree]]];
  imagesFinal[[t+1,3]]=ImageCrop[Rasterize[Rotate[ArrayPlot[RealWx[[t+3,5]],ColorFunction->RealWx[[t+3,2]],ColorFunctionScaling->False,Frame->False,  Axes->False
],rotationMorphed Degree]]];

  For[p=1,p<=3,p++,  (* Generate the prediction images P04, P104, P204 *)
                                          temp1=4-p;                          temp2=p;
   rotationMorphed= (  temp1*dataFirst[[4]] +       temp2*dataLast[[4]]  )/  temp3;  (* Currently, rotation isn't fully implemented *)
   dataMorphed=          (  temp1*dataFirst[[5]] +   temp2*dataLast[[5]]  )/temp3;  (* Image data for these should both be size x size *)
   (*dataMorphed=            Rescale[dataMorphed];*)
   white3=                          (  temp1*white1 +        temp2*white2 )/temp3;
   medgreen3=              (  temp1*medgreen1 +     temp2*medgreen2 )/temp3;
   darkgreen3=             (  temp1*darkgreen1 +      temp2*darkgreen2 )/temp3;
   yellow3=                        (  temp1*yellow1 +       temp2*yellow2 )/temp3;
   orange3=                      (  temp1*orange1 +       temp2*orange2 )/temp3;

   whichCF3=Function[z,
       If[z<#1,RGBColor[1,1,1],
        If[z<#2,If[CIWS==False,RGBColor[0,0.75,0],  GrayLevel[0.75]],
         If[z<#3,  If[CIWS==False,RGBColor[0,0.4,0],  GrayLevel[0.50]],
          If[z<#4,RGBColor[1,1,0],
           If[z<#5,RGBColor[1,0.5,0],  (* orange, the color pilots must avoid *)
            RGBColor[1,0,0] ]]]]]]&[white3,medgreen3,darkgreen3,yellow3,orange3] ;    (* red *)

   img=ArrayPlot[dataMorphed,ColorFunction->whichCF3,ColorFunctionScaling->False,Frame->False,  Axes->False ];
   imagesFinal[[t+1,p+3]]=ImageCrop[Rasterize[Rotate[img, rotationMorphed Degree]]];  (* Add the final prediction image, P4 *)
   ]; (* End For[p=1,p≤3,p++, *)
  (*tempData=Get[ StringJoin["P4[[",ToString[t+1],"]].dat"]];*)
  tempData=P4Frames[[t+1]];
   imagesFinal[[t+1,7]]=ImageCrop[Rasterize[Rotate[ArrayPlot[tempData[[5]],ColorFunction->tempData[[2]],ColorFunctionScaling->False,Frame->False,  Axes->False
],rotationMorphed Degree]]]
  ];    (* End For[  t=0,  t≤ 3,  t++, *)  (* Now, check the images, to make sure they came out all right before saving them *)
imagesFinal

(* Step 13--Save the finished set of images to file *)
AbsoluteTiming[Put[ imagesFinal, StringJoin[directoryName,".jpgs"]]]
 {4.98698,Null}
(* Step 14--Test the finished set of images  *)
AbsoluteTiming[Get[ StringJoin[directoryName,".jpgs"]]]
(* Exploring the future issue of how to determine point-of-closest-approach to a gnarly, fractal area. Use image-processing to eliminate all but the boundary between
orange and yellow ;*)
bimages=images;
perhaps=dataMorphed;
```

```
threshold=yellow;
SIZE=2;
TOLERANCE = 25; (* Must be 0 < TOLERANCE < 2 SIZE + 1 *)
lengthR=Length[dataMorphed[[1]]];
lengthC=Length[dataMorphed[[1,1]]];
Timing[
 For[n=1,n<=nFrames,n++,
  For[r=SIZE+1,r<= lengthR-SIZE,r++,
   For[c=SIZE+1,c<=  lengthC-SIZE,c++,
    If[ dataMorphed[[n,r,c]]>threshold,
     counter=0;
     For[i=r-SIZE,i<= r+SIZE,i++,
      For[j=c-SIZE,j<=  c+SIZE,j++,
       If[dataMorphed[[n,i,j]]>threshold,counter++]
       ]]];
    If[counter>= TOLERANCE, perhaps[[ n,r,c]]=0.1]
    ]]]]
For[i=1,i<=nFrames,i++,
  bimages[[i]]=ArrayPlot[perhaps[[i]],ColorFunction->whichCF2,ColorFunctionScaling->False,Frame->False,  Axes->False,(*ImageSize□{Scaled[.5]},*) ImageSize->{2^x+1 ,2^x+1
},AspectRatio->1/1]
  ];
GraphicsGrid[{{bimages[[1]],bimages[[2]],bimages[[3]]},{bimages[[4]],bimages[[5]],bimages[[6]]}},ImageSize->{3*(2^x+1) ,2*(2^x+1) }]
```

APPENDIX C

Part-Task Simulation *Mathematica* Code

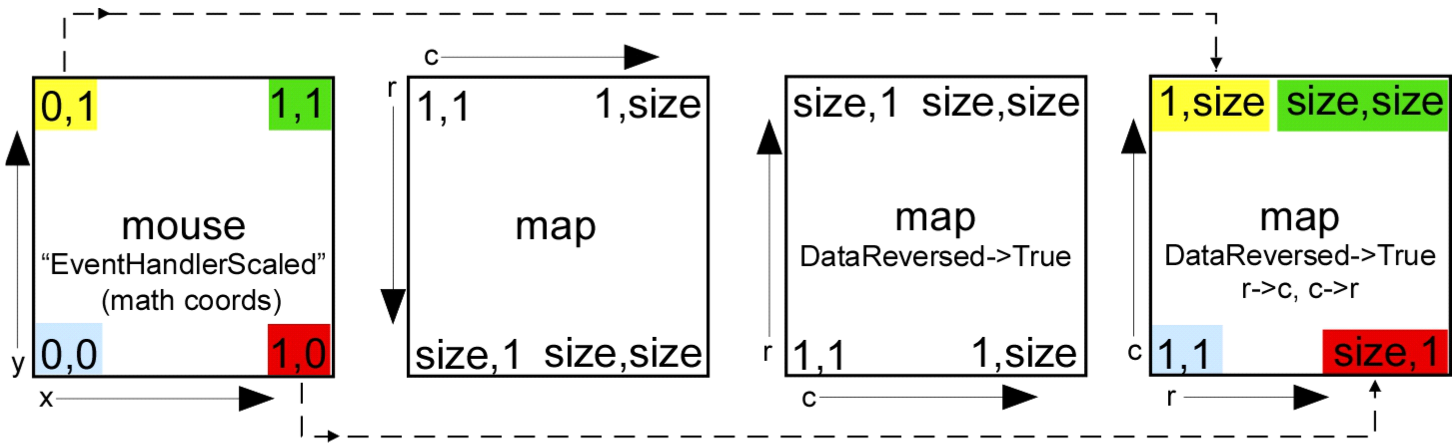
```
where="work";  (* Where am I coding, at home, or at work? *)
If[where=="home",SetDirectory["C:\\Billy Bob\\Telework\\PROBWX\\"], SetDirectory["C:\\Billy Bob\\Projects Too Big for M Drive\\PROBWX\\"]];

 sID=$Failed;  (* Input the user ID *)
While[sID==$Failed||sID=="",  (* Keep putting up ID box until a response is made *)
               sID=DialogInput[{res=""},
              Column[{"Enter your ID #", InputField[Dynamic[res],Number,ImageSize->{75,25}],
               Button["Begin",DialogReturn[res],ImageSize->Automatic]
               },Alignment->Center]
               ]
               ]; (* End While [id□*)

scenarios={{"H-SM","H-ST","H-LA"},{"D-SM","D-ST","D-LA"},{"P-SM","P-ST","P-LA"}};
Switch[Mod[sID,6],
  0,setOrder={1,2,3},
  1,setOrder={1,3,2},
  2,setOrder={2,3,1},
  3,setOrder={2,1,3},
  4,setOrder={3,1,2},
  5,setOrder={3,2,1}
  ];
finalOrder={};
For[i=1,i<=3,i++,
  members={1,2,3};
  chosenOrder={};
  For[j=1,j<=3,j++,
   numMembers=Length[members];
   whichMember=RandomInteger[{1,numMembers}];
   AppendTo[chosenOrder,scenarios[[setOrder[[i]],members[[whichMember]]]]];

   members=Drop[members,{whichMember}];
   ];
  AppendTo[finalOrder,chosenOrder];
  ];
finalOrder

numScenGrps=3;     numScens=3;       scenariosRemaining=numScenGrps*numScens;
 (* targetFPS=.7;Desired animation rate & refresh rate in frames per second *)
framesPerHr= 6;  (* Sampling rate for radar return *)
gTerrainSize={200,200};    (* {x,y} size of imaginary terrain-box over which we'll fly, in nm *)
gDisplaySize = {513,513};   (* Display image size,  {x, y} *)
rngRingRadius=20;   (* Radius of the range ring around the A/C icon *)
pixelsPerNM= N[gDisplaySize[[1]]/gTerrainSize[[1]] ]; (* # of pixels per nautical mile. In 2-hr flight @ 100 nm/hr □200 nm.  200 * 2.5 = 500 pixels representing 200 nm*)
                (* Initialize flight parameters *)
wxHdg=180;  (* Degrees aero, where N = 0, E = 90, etc *)
```

```
wxGrdSpdKT=30;  (* Kt *)
wxSpdPixelsPerFrame= N[wxGrdSpdKT*pixelsPerNM/framesPerHr];   (* How many pixels the wx moves per frame *)
wxHdgVector= N[ {Sin[wxHdg Degree], Cos[wxHdg Degree]}];
numFrameGrps=4; (* Number of 'frame groups'  per scenario (rows, one row of 7 images per 10 minutes of simulated time)  *)
gaugeSize={170,170};   (* How big to make the heading and speed gauges *)
gAC= Import["AC icon4.gif"];    gAC = ImageResize[gAC, {20, 20}];
(*;*)
toggleRR[rr_]:=Module[{},
  If[firstTimeRROn==True,
   AppendTo[rrData,{"STARTTIME", startTime  } ];
   firstTimeRROn=False
   ];
  If[rr==False,
   rngRing=True;
   AppendTo[rrData, {"ON",AbsoluteTime[] } ],
   rngRing=False;
   AppendTo[rrData, {"OFF",AbsoluteTime[] } ]
   ]
  ];

saveScenarioData[theData_,fileName_]:=Module[{outFile},
  defaultDirectory=NotebookDirectory[];
  outFile=defaultDirectory<>"Pilot data\\S"<>StringPadLeft[ToString[sID],2,"0"]<>"."<>folderName<>fileName<>".txt";
  ptr=OpenWrite[outFile];
  Write[ptr,theData];
  Close[outFile]
  ]; (* End saveScenarioData *)

saveAllData:=Module[{},
  t$acPosn$Hdg$Spd$wxPosn[[11,1]]=AbsoluteTime[];  (* Time this scenario ended *)
  saveScenarioData[t$acPosn$Hdg$Spd$wxPosn,".main"]; (* Save the primary data structure *)
  saveScenarioData[rrData,".ring"];  (*Save the range-ring data *)
  ];
                              (* Below,11 rows, representing 10 time slices at 10-minute intervals, plus a last row containing when the pilot hit the "Next scenario" button *)
(*;*)
advance[]:=Module[ {},
  t$acPosn$Hdg$Spd$wxPosn[[nowRowIndex,1]]=AbsoluteTime[];  (* Record when the 10-minute time advance was made.*)
  acStartPosnNM=t$acPosn$Hdg$Spd$wxPosn[[nowRowIndex,2]];
  wxStartPosnNM=t$acPosn$Hdg$Spd$wxPosn[[nowRowIndex,5]];
  For[r=1,r<= 4,r++, (* Initialize A/C and wx position (nm)/heading (degrees aero)/speed (kt) for the next 7 frames, t = -30, -20...20, 30 *)
        t$acPosn$Hdg$Spd$wxPosn[[r+nowRowIndex,2]]=acStartPosnNM+(r* (acGrdSpdKT*acHdgVector/framesPerHr)) ;
        t$acPosn$Hdg$Spd$wxPosn[[r+nowRowIndex,3]]=acHdg; (* A/C heading, degrees, in aero coords, N=0, E=90, etc. *)
        t$acPosn$Hdg$Spd$wxPosn[[r+nowRowIndex,4]]=acGrdSpdKT; (* knots *)
        t$acPosn$Hdg$Spd$wxPosn[[r+nowRowIndex,5]]=wxStartPosnNM+(r*(wxGrdSpdKT*wxHdgVector/framesPerHr))
    ];
  currentSimET+=10;
  frmGrp++;
  If[frmGrp==numFrameGrps,advButtonText="Maneuvers finished";nextScenButtonText="Go to next scenario"];
  nowRowIndex++;  (* "Now" began at row 4, and advances 1 every time the "Fly 10 minutes" advance button is pressed *)
  currentBackgrd=ImageCompose[origBackgrd,Graphics[{Orange, Disk[ImageScaled[Round[t$acPosn$Hdg$Spd$wxPosn[[nowRowIndex, 2]] *pixelsPerNM] /gDisplaySize[[1]]
```

```
],0.015]}]];
   Refresh[frmGrp]
   ]; (* End Module advance *)

gauge1=HorizontalGauge[Dynamic@frameTime,{-30,30},ScaleDivisions->7,GaugeLabels->"Time" ];  (* This is the horizontal 'slider' *)
               (* Below, operating on gauge2 or 3 will trigger a reset of the Animate "frame" variable to 1. That's what TrackedSymbols□{whatever} is used for *)
gauge2 = AngularGauge[Dynamic@acHdg, {0, 359}, ScaleOrigin -> {Pi/2, -3 Pi/2 }, ScaleDivisions -> {18, 2}, GaugeMarkers -> "FlatArrowNeedle", ImageSize->gaugeSize];
gauge3 = AngularGauge[Dynamic@acGrdSpdKT, {0, 200}, ScaleOrigin -> {Pi/2, -3 Pi/3 }, ScaleDivisions -> {18, 2}, GaugeMarkers -> "ScaleRange", GaugeLabels->"Ground\nSpeed",ScaleRanges -> {{0, 80} -> Pink,     {80, 140} -> Green, {140, 200} -> Red}, ImageSize->gaugeSize ];
limiterUp1:=If[frmGrp<numFrameGrps,acHdg++];               limiterUp5:=If[frmGrp<numFrameGrps,acHdg+=5];
limiterDown1:=If[frmGrp<numFrameGrps,acHdg--];       limiterDown5:=If[frmGrp<numFrameGrps,acHdg-=5];

upOne=ClickPane[Dynamic@Graphics[{ Red,Triangle[{{0,0},{15,15},{30,0}},VertexColors->{Black,Red,Green}] },          ImageSize->{30,15}],limiterUp1&];upFive=ClickPane[Dynamic@Graphics[{ Black,Rectangle[{0,-15},{30,0},RoundingRadius->2] },   ImageSize->{30,15}],limiterUp5&];
downOne=ClickPane[Dynamic@Graphics[{ Red,Triangle[{{0,0},{15,-15},{30,0}},VertexColors->{Green,Red,Black}] },ImageSize->{30,15}],limiterDown1&];downFive=ClickPane[Dynamic@Graphics[{ Black,Rectangle[{0,-15},{30,0},RoundingRadius->2] } ImageSize->{30,15}],limiterDown5&];
advanceButton = Dynamic[Button[advButtonText, If[frmGrp<  numFrameGrps,  advance[ ] ], ImageSize -> Automatic ]];
rangeRingButton=Button["Toggle range ring",toggleRR[rngRing],ImageSize->Automatic];
nextScenButton = Dynamic[Button[nextScenButtonText, DialogReturn[22], ImageSize -> Automatic]];

currentET =Row[{Text[Style["Current simulation time: ",14,Red,FontFamily->"Arial"  ]],      Dynamic[Text[Style[ToString[currentSimET], 20,Red,FontFamily->"Arial"] ]]}];
(*textFPS =Row[{Text[Style["Current frame rate: ",14,Red,FontFamily□"Arial"  ]],      Dynamic[Text[Style[ToString[averageFPS], 20,Red,FontFamily□"Arial"] ]]}];*)

drawSim=Dynamic@Row[ {  (* This will get called every time Animate loops *)
                             Column[{ Row[{Magnify[mScale,1.6] ,Spacer[30]}],
                          (*Row[{gauge1,(*Text["time"]*)  Spacer[50]}],*)
                          Row[ {gauge2, Column[ {  Column[ {upOne, upFive},Center,0],
                                                               Style[ToString[acHdg = Mod[Round[acHdg] ,360]],Bold,15],
                                                                Column[{downFive,downOne},Center,0] },Center,0.25
                                                               ]
                             }],  (* End inner Row *)
                                              Row[ {gauge3,  ToString["   "] }],
                                              advanceButton,
                         currentET,
                      (*   textFPS,*)
                                   rangeRingButton,
                                  Spacer[20],
                        nextScenButton
                                            },Alignment->Center], (* End outer Column *)
     Show[currentFrame, ImageSize -> gDisplaySize, Frame -> False, FrameTicks -> None]
       }] ;   (*,  End outer Row *)

For[scengrp=1,scengrp<=numScenGrps,scengrp++,
  For[scen=1,scen<=numScens,scen++,
   acPosnNM =acStartPosnNM= {Round[gTerrainSize[[1]]/2], 60};  (* Starting point of the aircraft icon, {x, y, in nm}, AT TIME = 0, w {0,0} = the lower left-hand corner  *)
   acGrdSpdKT =acOldGrdSpdKT= 110; (*  Aircraft speed, kt*)
   acSpdPixelsPerFrame= N[acGrdSpdKT*pixelsPerNM/framesPerHr];   (* How many pixels the A/C moves per frame *)
   acOldHdg = acHdg = 0; (* Aircraft heading at t = 0, degrees aeronautical, 0 = north, increasing clockwise *)
   acHdgVector= N[ {Sin[acHdg Degree], Cos[acHdg Degree]}];
   wxStartPosnNM= {Round[gTerrainSize[[1]]/2], 190};  (* 200, Starting point of the weather cell at time = -30, {x, y in nm}, w {0,0} being the lower left-hand corner of the image *)
```

```
   rngRing=False;   (* Toggle 20 nm range ring on or off *)
   firstTimeRROn=True;  (* a flag used to capture the first time the range ring is turned on *)
   t$acPosn$Hdg$Spd$wxPosn=Table[0,{rows,11},{cols,5}]; (*columns holding TIME, A/C POSITION, (x,y, in nm), A/C HEADING (deg aero), A/C SPEED (kt), WX POSITION, at each
of the 11 rows in time *)
   currentSimET=0;  (* Current simulation pseudo-elapsed time (NOT absolute time) *)
   nowRowIndex=4; (* Which frame represents time row = 0 *)
   advButtonText="Fly for 10 minutes";
   scenariosRemaining--;
   If[scenariosRemaining>= 1,nextScenButtonText=StringJoin[ToString[scenariosRemaining]," scenario(s) remaining"],nextScenButtonText="Last scenario"];
   frmGrp=1; rrData={};

   If[where=="home",SetDirectory["C:\\Billy Bob\\Telework\\PROBWX\\"], SetDirectory["C:\\Billy Bob\\Projects Too Big for M Drive\\PROBWX\\"]];
          folderName= finalOrder[[scengrp,scen]];
         Switch[ Characters[folderName][[1]],
    "H", numFrames=3;
             mScale=Import["Manipulator scale H.jpg"];
             origBackgrd=currentBackgrd=Import["backgroundWScaleH.jpg"];
    "D", numFrames = 7; (* Number of graphical frames we'll display per loop (one row of frames) *)
            origBackgrd=currentBackgrd=Import["backgroundWScaleD.jpg"];
            mScale=Import["Manipulator scale.jpg"];
    "P", numFrames = 7; (* Number of graphical frames we'll display per loop (one row of frames) *)
            origBackgrd=currentBackgrd=Import["backgroundWScaleP.jpg"];
            mScale=Import["Manipulator scale.jpg"];
    ];

      defaultDurn=1.5*numFrames;        refreshRate=3;
       If[where=="home",SetDirectory["C:\\Billy Bob\\Telework\\PROBWX\\"<>folderName],  SetDirectory["C:\\Billy Bob\\Projects Too Big for M Drive\\PROBWX\\"<>folderName]];
   AbsoluteTiming[gWx=Get[ folderName<>".jpgs" ]];
   startTime=t$acPosn$Hdg$Spd$wxPosn[[1,1]]=AbsoluteTime[]; (* t0 = Official time that this scenario started *)
       For[r=-3,r<= 3,r++, (* Initialize A/C and wx position (nm)/heading (degrees aero)/speed (kt) for first 7 frames, t = -30, -20...20, 30 *)
          t$acPosn$Hdg$Spd$wxPosn[[r+nowRowIndex,2]]=acStartPosnNM+(r* (acGrdSpdKT*acHdgVector/framesPerHr)) ;
          t$acPosn$Hdg$Spd$wxPosn[[r+nowRowIndex,3]]=acHdg; (* A/C heading, degrees, in aero coords, N=0, E=90, etc. *)
          t$acPosn$Hdg$Spd$wxPosn[[r+nowRowIndex,4]]=acGrdSpdKT; (* knots *)
          t$acPosn$Hdg$Spd$wxPosn[[r+nowRowIndex,5]]=wxStartPosnNM+(r*(wxGrdSpdKT*wxHdgVector/framesPerHr))
     ];   (* r+3 *)
                                                                                (* Below, the orange disk indicates the t = 0 (current) position of the aircraft *)
   currentBackgrd=ImageCompose[origBackgrd,Graphics[{Orange, Disk[ImageScaled[Round[t$acPosn$Hdg$Spd$wxPosn[[nowRowIndex=4, 2]] *pixelsPerNM] /gDisplaySize[[1]]
],0.015]}]];
   oldFrame=0;  (* Frame-counter, to used to trigger computational updates, and to estimate Animate's actual frames-per-second*)

   (*out=*)DialogInput[Animate[   (* NOTE: Animate may cycle multiple times before incrementing its control variable, "frame" *)
     If[ acHdg!= acOldHdg || acGrdSpdKT != acOldGrdSpdKT,  (* IF a maneuver's taken place, update hdg and spd  *)
      t$acPosn$Hdg$Spd$wxPosn[[+nowRowIndex,3]]= acOldHdg=acHdg;                    acHdgVector= N[ {Sin[acHdg Degree], Cos[acHdg Degree]}]; (* {x,y} heading vector, aero
coorss (N-up, 0 @ 12 o'clock *)
      t$acPosn$Hdg$Spd$wxPosn[[+nowRowIndex,4]]=acOldGrdSpdKT=acGrdSpdKT;
      For[r=nowRowIndex+1,r<= nowRowIndex+3,r++, (* modify A/C position (nm)/heading (degrees aero)/speed (kt) for all frames 5 to 7 *)
       t$acPosn$Hdg$Spd$wxPosn[[r,2]]=t$acPosn$Hdg$Spd$wxPosn[[r-1,2]]+(acGrdSpdKT*acHdgVector/framesPerHr) ;
       t$acPosn$Hdg$Spd$wxPosn[[r,3]]=acHdg; (* A/C heading, degrees, in aero coords, N=0, E=90, etc. *)
       t$acPosn$Hdg$Spd$wxPosn[[r,4]]=acGrdSpdKT (* knots *)
        ]
```

```
     ];  (* End  acHdg!= acOldHdg || acGrdSpdKT != acOldGrdSpdKT,  *)
    If[frame!= oldFrame,  (* Only compute this stuff & draw once per frame. Otherwise, Animate tries to cycle so fast it crashes *)
     frameTime=(10 frame)-40;  (* The simulated time of each frame, in minutes *)
     oldFrame=frame;  (* Maneuvers only affect time = now, which is t$acPosn$Hdg$Spd$wxPosn[[r,4]] *)

     acXYPosnPixels = Round[t$acPosn$Hdg$Spd$wxPosn[[frame+(nowRowIndex-4), 2]] *pixelsPerNM] ;
     acSpdPixelsPerFrame =    t$acPosn$Hdg$Spd$wxPosn[[frame+(nowRowIndex-4),4]]*pixelsPerNM;
     wxXYPosnNM=t$acPosn$Hdg$Spd$wxPosn[[frame+(nowRowIndex-4),5]];
     wxXYPosnPixels=Round[wxXYPosnNM*pixelsPerNM];     (* Move the weather *)

     currentFrame = ImageCompose[currentBackgrd,gWx[[frmGrp,frame]],wxXYPosnPixels ]  ; (*  Add the weather. {0,0} is in lower-lefthand corner.  Form→ {image, overlay, position}
*)
     acIconAtHdg = ImageRotate[gAC, -t$acPosn$Hdg$Spd$wxPosn[[frame+(nowRowIndex-4),3]] Degree];  (* Rotate the A/C icon, then add it. Have to "minus" acHdg because
ImageRotate > 0 moves counterclockwise *)
     currentFrame = ImageCompose[currentFrame, acIconAtHdg, acXYPosnPixels];  (* Add the A/C icon *)
     If[rngRing== True,
      currentFrame = ImageCompose[currentFrame, Graphics[Text[Style[ToString[frameTime], 20,  Blue]]], acXYPosnPixels+{0,20}];   (* Add sim frame time, -30, -20, etc, near the A/C
icon *)
      currentFrame = ImageCompose[currentFrame, Graphics[Circle[ Offset[acXYPosnPixels-{256,256}],ImageScaled[  0.12  ]]]],  (* Add 20 nm range ring around the A/C icon *)
      currentFrame = ImageCompose[currentFrame, Graphics[Text[Style[ToString[frameTime], 20,  Blue]]], acXYPosnPixels+{0,20}]
      ]
     ];   (* End If[frame ≠ oldFrame *)       (* ^^^MMCA takes the center of the frame as {0,0} of its coordinate system *)
    drawSim,
          {frame,1,numFrames,1,AppearanceElements-> {"ProgressSlider","PlayPauseButton"(*,"FasterSlowerButtons"*),"StepLeftButton","StepRightButton"} ,ImageSize->{200,20}},
    AnimationRunning->True, RefreshRate->refreshRate,(*AnimationRate->targetFPS, *) DisplayAllSteps->True, ContinuousAction->True, DefaultDuration->defaultDurn,
    TrackedSymbols->{frmGrp,acGrdSpdKT,acHdg,rngRing} ,LocalizeVariables->False(*, Deinitialization□(saveAllData)*)
    ],ImageSize->Automatic
   ]; (* End DialogInput *)
  saveAllData
  ]]; (* End For[scengrp,   For[scen    loops *)
```



APPENDIX D

Instructions Given to Pilots

THANKS

The FAA's Weather Technology in the Cockpit Program (WTIC Program) and Civil Aerospace Medical Institute (CAMI) want to take this opportunity to thank you for agreeing to participate in this study. Without the help of pilots like you, we couldn't do research like this.

The reason for this experiment is to see what you think about 3 kinds of weather display. It's **not** a test. There's no "pass" or "fail." No grades, and nothing goes into your Airman Record. Your data will be strictly anonymous forever. You'll get to practice before we actually start collecting data. You'll learn interesting things about future technology, and make money doing it. So, relax, have fun. welcome aboard.

WHAT WE'LL DO TODAY

1. First fill out the **Informed Consent Form.**
2. Second, take a copy of the **Pay Form** home with you. You'll need that to get paid.
3. Third, study the information below (**BACKGROUND**, etc). Take as long as you want. Ask questions. We like questions.
4. Fourth, we'll do **practice runs** before each block of 3 scenarios. Practice as long as you want. Ask questions.
5. Fifth—when you feel ready—we'll **start the experiment**.
6. Last, we'll have a short debrief session at the end of each block of 3 scenarios.

BACKGROUND

OK, here we go. This study examines how pilots might understand and use **looping weather displays**. This requires some background explanation.

There are 3 main kinds of looping weather displays (and you'll see all 3 in this experiment):

Historical Displays pictures of *recent weather* (e.g., regular looping NEXRAD)
Deterministic Displays pictures of *predicted weather*, as if there were only *one way* the future could turn out. More on this in a minute.
Probabilistic Displays pictures of *predicted weather*, as if there were *many ways* the future could turn out, and we're not quite sure which one will actually happen. More on this, too, in a minute.

#1—Historical Weather Radar Displays

OK, we all know looping NEXRAD, so #1 is easy. Standard looping NEXRAD is technically called a "**historical**" display because it shows us a "movie about history", about weather that already happened.

Historical displays do NOT predict future weather.

Now, let's practice what this is going to look like in our experiment.

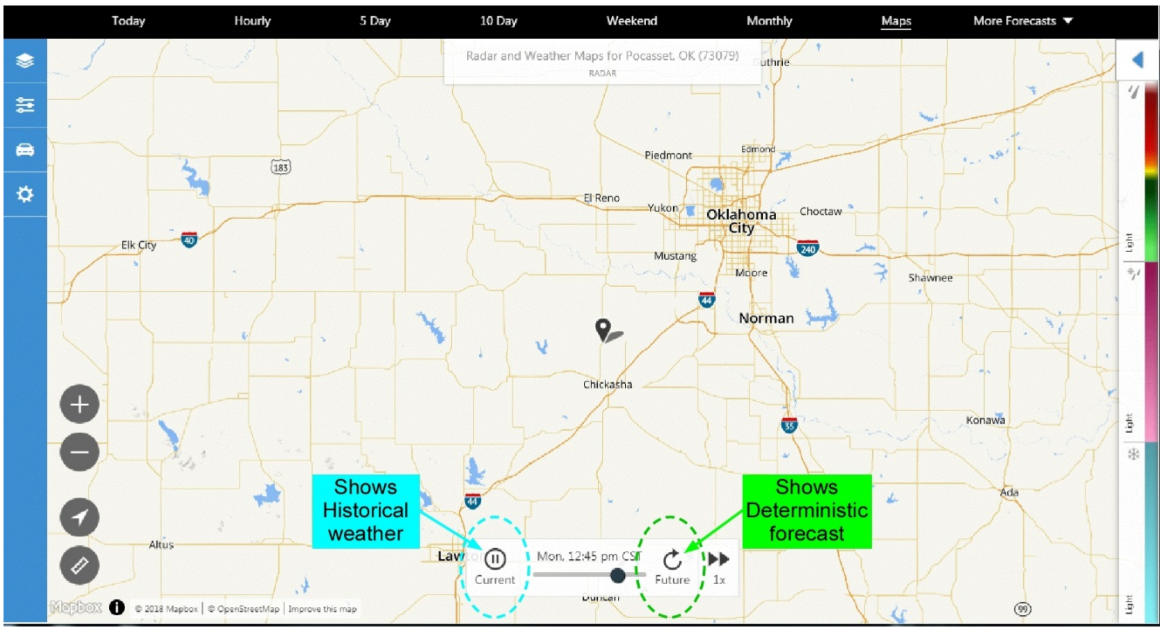
#2—Deterministic Weather Radar Displays

Always striving to do better, meteorologists are developing new kinds of **computer models** that calculate and display forecast weather. In fact, there are dozens of various kinds of weather models.

A *model* is *a simplification of a complex real system, for example, a storm. A good model behaves a lot like the thing it's imitating, but is a lot easier to work with*. The important things to remember are: 1) No model is ever completely accurate, and; 2) The farther into the future a model tries to predict, the more error there is..

Now, recall that #2 on our list was the "**deterministic"** weather display. We'll define that word later. For now, know that a deterministic display shows us the results of one computer weather model run one single time.

You may have seen a deterministic forecast if you clicked on the "Future" option on some NEXRAD displays (although it's not really NEXRAD itself creating that forecast, it's a separate computer program).



Some facts about a deterministic weather forecast, and why it's called "deterministic":

a) Only one, single weather model is run only one, single time.
b) The starting values (for temperature, pressure, wind speed, etc) plugged into that model are based on just one set of readings taken from different places, AND those readings are, at best, about 10 minutes old (5-15, so 10 is an average)..
c) So, **the model is incomplete, & will get less and less accurate, the father into the future we try to predict**.
d) Now, here's why it's called "deterministic": Just like the rules of arithmetic say that "2+2 is determined to be 4"—and no other number—computers run on determined rules, too. So, we say they're **deterministic**, and so are the programs we run on them. No matter how many times we run a computer program, as long as the starting numbers stay the same, the rules determine we'll get the exact same results every time. And, this is exactly how weather forecasts work.
e) **However, that deterministic forecast will never be 100% correct, because the model it's running never is**.

These are critical concepts you need to know to understand how deterministic weather forecasts work. Now, let's see how these actually work in our experiment.

#3, and last on our list, is the "**probabilistic**" forecast. A probabilistic forecast is a kind of rating system that rates future weather in terms of **"how likely is something to happen?"** For instance, a fair coin toss is 50% likely to be heads.

Unlike a deterministic forecast, **a probabilistic weather forecast is based on multiple weather model runs**. We can either run a single weather model many times—slightly varying the initial wind speeds, temperatures, etc., each time—or, we can run a dozen or more different weather models, one or more times each.

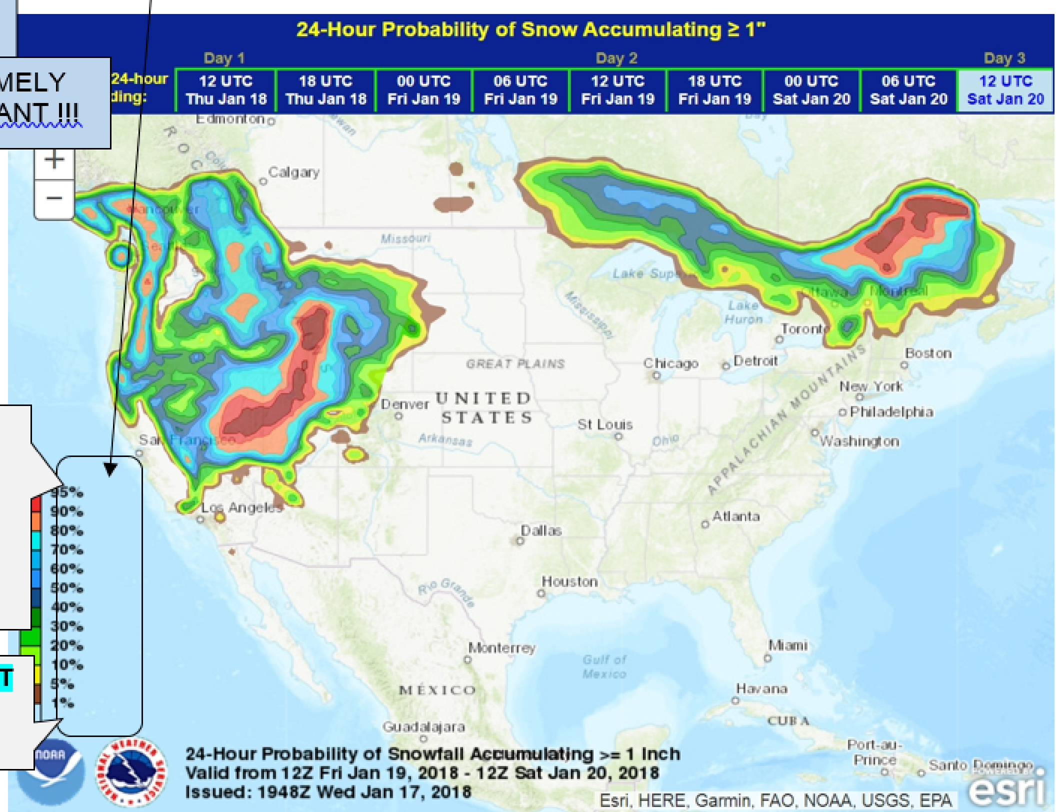
The point is that a probabilistic forecast first **generates a range of different results**, and then **calculates what percentage of those results agree for every place on the map**. **The bigger that percentage, the more certain we are that that forecast will be correct for that location at that time.**

Now, all this may be a bit new and confusing, so let's look at a concrete example.

Below is a picture of a National Weather Service probabilistic forecast. This shows how likely it is to snow **at least 1 inch in any given place**. Notice the **scale** on the left-hand side of the map. This scale runs from **1 to 95% probability**. So, for example, the darkest-red color means that—for that particular region on the map—95% of their models agreed that there'd be 1 or more inches of snow."

☞ Note: This does NOT tell us **HOW MANY** inches are predicted—only "**the chance of at least 1 inch**." ☜

EXTREMELY IMPORTANT !!!



This is their probability scale.

95% means "ALMOST CERTAINLY 1 inch of snow or more."

1% means "ALMOST NO CHANCE of 1 inch or more."

3 of today's 9 weather scenarios will be probabilistic. In those 3 probabilistic scenarios—**color shows the percent chance of encountering <u>worse than 40 dBZ</u> radar reflectivity**—where the colors **orange** and **red** represent the composite reflectivity for heavy weather that Advisory Circular AC 00-24-C advises us to avoid.

Probabilistic weather may seem kind of abstract. So, let's practice what this is going to look like in our experiment, and this should help to show how theory translates into practice.

BASIC PLAN

1. Imagine you're flying an **a GA aircraft you're well-familiar with**.
2. Initial cruise airspeed will be **110 kt**, But you're free to change that at any point.
3. **We're not going to worry about altitude changes** in this experiment. Heading and airspeed only.
4. You'll "fly" **9 weather scenarios**.
5. Cockpit weather display will show **looping weather radar**, one type per scenario, as we said

a. Historical only
b. Historical + deterministic
c. Historical + probabilistic

6. You can **stop the looping** any time you want. And, you can use the **slider** to "**move back and forth in time**," if it helps you plan the best course.
7. You can turn on a **range ring** if you prefer. And, we are interested in how much you use that, and how helpful you feel it is.

Your job is to **bear north (up),** as much as possible, while **maintaining at least 20 nm separation from hazardous weather**.

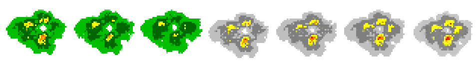
APPENDIX E

Debrief Questions

Today you saw 3 different kinds of weather display.

Time (minutes)						
Past			Now	Future		
-30	-20	-10	0	10	20	30

_____Participant ID #

1. **Historical** display: 
2. **Deterministic** display: 
3. **Probabilistic** display: 

As you answer these questions, please be honest, because that's how good science gets done. Tell us what you really think—not what you think we might want to hear. Keep in mind that all this is completely confidential. As these sheets are tallied, nobody will see your name—only an ID number—and absolutely nothing goes in your Airman Record.

1. How much did you like each type of weather display? (circle one number per question)

	Not at all			Neutral			Very much
a. Historical	1	2	3	4	5	6	7
b. Deterministic	1	2	3	4	5	6	7
c. Probabilistic	1	2	3	4	5	6	7

2. How much mental effort did it take you to use each type of display?

	A lot			Average			Very little
a. Historical	1	2	3	4	5	6	7
b. Deterministic	1	2	3	4	5	6	7
c. Probabilistic	1	2	3	4	5	6	7

3. How much training and practice would each type of display require to for you get really good at it?

	A lot			Average			Very little
a. Historical	1	2	3	4	5	6	7
b. Deterministic	1	2	3	4	5	6	7
c. Probabilistic	1	2	3	4	5	6	7

4. In actual hazardous-weather flight, how safe do you feel each display could keep you?

	Not at all safe			Average			Very safe
a. Historical	1	2	3	4	5	6	7
b. Deterministic	1	2	3	4	5	6	7
c. Probabilistic	1	2	3	4	5	6	7

5. How much do you think <u>other pilots</u> would like each type of display?

	Not at all			Neutral			Very much
a. Historical	1	2	3	4	5	6	7
b. Deterministic	1	2	3	4	5	6	7
c. Probabilistic	1	2	3	4	5	6	7

6. How much mental effort do you figure it would take <u>other pilots</u> to use each type of display?

	A lot			Average			Very little
a. Historical	1	2	3	4	5	6	7
b. Deterministic	1	2	3	4	5	6	7
c. Probabilistic	1	2	3	4	5	6	7

7. How much training and practice would each type of display require for <u>other pilots</u> to get really good at it?

	A lot			Average			Very little
a. Historical	1	2	3	4	5	6	7

		1	2	3	4	5	6	7
b.	Deterministic	1	2	3	4	5	6	7
c.	Probabilistic	1	2	3	4	5	6	7

8. In actual hazardous-weather flight, how safe do you feel each display could keep <u>other pilots</u>?

		Not at all safe			Average			Very safe
a.	Historical	1	2	3	4	5	6	7
b.	Deterministic	1	2	3	4	5	6	7
c.	Probabilistic	1	2	3	4	5	6	7

9. How helpful was the range ring in keeping you separated from the weather in each of the displays?

		Useless			Fairly helpful			Essential
a.	Historical	1	2	3	4	5	6	7
b.	Deterministic	1	2	3	4	5	6	7
c.	Probabilistic	1	2	3	4	5	6	7

10. How much of a safety bias, or "secret safety buffer" do you suspect the National Weather Service might build into the deterministic and probabilistic displays, once they finally start being commonplace? (We left out the historical category, since we know that NEXRAD is nothing but radar data, no models involved).

		A lot			Some			None
a.	Deterministic	1	2	3	4	5	6	7
b.	Probabilistic	1	2	3	4	5	6	7

11. In your own words, describe what <u>information</u> we can get from a historical display such as NEXRAD. (For instance, what do the colors mean? What benefit does looping the images add?)

12. What information can we get from a deterministic display? (What do its colors mean? How is it the same as the historical display? What extra benefit does it give us?)

13. What information can we get from a probabilistic display? (What do its colors mean? How is it the same as the other two? What extra does it give us?)

14. What about any of the displays felt confusing to you (or you suspect would confuse other pilots)? (Don't hold back. You won't hurt anybody's feelings. Remember that this is just science. It's all about data. And <u>your</u> opinion IS data.)

15. What level of risk would <u>you</u> figure acceptable to fly through on a probabilistic display? ____%
(Give a number from 0-100%, meaning "the percent chance of my encountering 40 dBZ composite reflectivity, or worse.
So, "0%" would mean "I wouldn't want to ever take any risk under any circumstances whatsoever,"
"50%" would mean "I'd accept a 50-50 chance of running into heavy weather."
and "100%" would mean "I'd completely disregard the weather display and fly anywhere, anytime.")