84-GNC-49
Highlights from the FAA Support of the National Airspace System

Paper AIAA-2009-6077:
"An Algorithmic Method for Regression Analysis of Conflict Probe Accuracy"

Presented to: AIAA Guidance, Navigation, and Control

Authors: Andrew Crowell and Confesor Santiago

Date: August 12, 2009

Federal Aviation Administration

Overview

- Background of Conflict Probes
- Conflict Probe Accuracy
- Regression Analysis of Conflict Probes
- General Problem Definition
- Pairing Algorithm
- Conflict Pairing Program
- Examples
- Questions

Background of Conflict Probes

- **A Conflict Probe (CP) is a Decision Support Tool used by Air Traffic Controllers**
- **Assists in air traffic separation management**
- **Creates 4D(lat, lon, alt, time) trajectory predictions**
- **Generates conflict alerts**
  - Conflict occurs when there is a loss of separation between aircraft (usually 5nm horz, 1000 ft vert)

Conflict Probe Accuracy

- **Two areas of accuracy analysis**
  - Trajectory Accuracy Analysis
    - Compares the 4D trajectory predictions to the actual track of the aircraft
    - Analyzes spatial and temporal coincidence
  - Conflict Accuracy Analysis
    - Compares generated alerts to actual conflicts occurred
    - Analyzes temporal coincidence and alert properties

Regression Analysis

- **To help analyze the accuracy of a CP, the results are compared to another CP**
  - Current Operational CP (URET)
  - Previous version of itself
- **Investigate areas where a CP degrades in performance**
- **Two previous methods of regression analysis**
  - Statistical approach
  - Manually comparing

Only provides high-level performance analysis

Time consuming

Algorithm was needed to efficiently generate low-level analysis

General Problem Definition

- **Two arrays (A and B) with common attributes (same flight pair)**
- **Each entry in A may have similar quantifiable properties to an entry in B (such as temporal coincidence)**
- **Each entry in A may have similar properties to *multiple* entries in B, and vice versa**

A	a1	a2	a3	...
B	b1	b2	b3	...

Find the best match, if any, for each entry in A to a single entry in B, coincidentally finding the best match for each entry in B.

Pairing Algorithm

Definitions

- ***pair*** **– a single entry from array A compared to a single entry from array B**
- ***score*** **– the quantifiable value of the quality of a *pair***
- ***f(a, b)*** **– the function that determines the *score* of a *pair***
- ***match*** **– the *pair* with the best *score***

Pairing Algorithm (cont.)

A
7
12
5
9

B
5
4
11
7
3

$$f(a, b) = |a_x - b_y|$$

Create a matrix from the arrays, using f(a, b) as the elements, and search for the highest

$$|7 - 5| = 2$$

		B				
		[1] 5	[2] 4	[3] 11	[4] 7	[5] 3
A	[1] 7	2	3	4	0	4
	[2] 12	7	8	1	5	9
	[3] 5	0	1	6	2	2
	[4] 9	4	5	2	2	6

Pairing Algorithm (cont.)

Search the first row for the best score.

		B				
		[1] 5	[2] 4	[3] 11	[4] 7	[5] 3
A	[1] 7	2	3	4	0	4
	[2] 12	7	8	1	5	9
	[3] 5	0	1	6	2	2
	[4] 9	4	5	2	2	6

Pairing Algorithm (cont.)

Search the first row for the best score.

This element is called the pivot element.

		B				
		[1] 5	[2] 4	[3] 11	[4] 7	[5] 3
A	[1] 7	2	3	4	0	4
	[2] 12	7	8	1	5	9
	[3] 5	0	1	6	2	2
	[4] 9	4	5	2	2	6

The best score found is 4, but there is a tie.

In a tie, the leftmost score found is used.

Pairing Algorithm (cont.)

Search the column of the pivot element for the best score

		B				
		[1] 5	[2] 4	[3] 11	[4] 7	[5] 3
A	[1] 7	2	3	4	0	4
	[2] 12	7	8	1	5	9
	[3] 5	0	1	6	2	2
	[4] 9	4	5	2	2	6

The best score found is 6.

This is the new pivot element.

Pairing Algorithm (cont.)

Search the row of the pivot element for the best score

		B				
		[1] 5	[2] 4	[3] 11	[4] 7	[5] 3
A	[1] 7	2	3	4	0	4
	[2] 12	7	8	1	5	9
	[3] 5	0	1	6	2	2
	[4] 9	4	5	2	2	6

Matches

$a_3(5) – b_3(11)$

The best score does not change from the previous iteration

This is the match

Pairing Algorithm (cont.)

Since a match is found, remove both the column and the row of the match

		B			
		[1] 5	[2] 4	[4] 7	[5] 3
A	[1] 7	2	3	0	4
	[2] 12	7	8	5	9
	[4] 9	4	5	2	6

Matches

$a_3(5) – b_3(11)$

This is done because each entry can only be matched once

Pairing Algorithm (cont.)

Repeat the process, searching the first row

		B			
		[1] 5	[2] 4	[4] 7	[5] 3
A	[1] 7	2	3	0	4
	[2] 12	7	8	5	9
	[4] 9	4	5	2	6

Matches

$a_3(5) – b_3(11)$

This process continues until all elements have been removed from the matrix

Pairing Algorithm (cont.)

Repeat the process

		B			
		[1] 5	[2] 4	[4] 7	[5] 3
A	[1] 7	2	3	0	4
	[2] 12	7	8	5	9
	[4] 9	4	5	2	6

Matches

$a_3(5) – b_3(11)$

This process continues until all elements have been removed from the matrix

Pairing Algorithm (cont.)

Repeat the process

		B			
		[1] 5	[2] 4	[4] 7	[5] 3
A	[1] 7	2	3	0	4
	[2] 12	7	8	5	9
	[4] 9	4	5	2	6

Matches

$a_3(5) – b_3(11)$

$a_2(12) – b_5(3)$

This process continues until all elements have been removed from the matrix

Pairing Algorithm (cont.)

Repeat the process

		B		
		[1] 5	[2] 4	[4] 7
A	[1] 7	2	3	0
	[4] 9	4	5	2

Matches

$a_3(5) – b_3(11)$

$a_2(12) – b_5(3)$

This process continues until all elements have been removed from the matrix

Pairing Algorithm (cont.)

Repeat the process

		B		
		[1] 5	[2] 4	[4] 7
A	[1] 7	2	3	0
	[4] 9	4	5	2

Matches

$a_3(5) – b_3(11)$

$a_2(12) – b_5(3)$

This process continues until all elements have been removed from the matrix

Pairing Algorithm (cont.)

Repeat the process

		B		
		[1] 5	[2] 4	[4] 7
A	[1] 7	2	3	0
	[4] 9	4	5	2

Matches

$a_3(5) – b_3(11)$

$a_2(12) – b_5(3)$

This process continues until all elements have been removed from the matrix

Pairing Algorithm (cont.)

Repeat the process

		B		
		[1] 5	[2] 4	[4] 7
A	[1] 7	2	3	0
	[4] 9	4	5	2

Matches

$a_3(5) – b_3(11)$

$a_2(12) – b_5(3)$

$a_4(9) – b_2(4)$

This process continues until all elements have been removed from the matrix

Pairing Algorithm (cont.)

Repeat the process

		B	
		[1] 5	[4] 7
A	[1] 7	2	0

Matches

$a_3(5) – b_3(11)$

$a_2(12) – b_5(3)$

$a_4(9) – b_2(4)$

This process continues until all elements have been removed from the matrix

Pairing Algorithm (cont.)

Repeat the process

		B	
		[1] 5	[4] 7
A	[1] 7	2	0

Matches

$a_3(5) – b_3(11)$

$a_2(12) – b_5(3)$

$a_4(9) – b_2(4)$

$a_1(7) – b_1(5)$

This process continues until all elements have been removed from the matrix

Pairing Algorithm (cont.)

Repeat the process

Matches

$a_3(5) – b_3(11)$

$a_2(12) – b_5(3)$

$a_4(9) – b_2(4)$

$a_1(7) – b_1(5)$

None – $b_4(7)$

This process continues until all elements have been removed from the matrix

Conflict Pairing Program

- **Compares two CPs to match each Conflict Prediction in CP A to one in CP B**
- **Developed in Java using Object Oriented Design principals**
- **High level algorithm is abstracted from scoring formula**
- **OOD allows exploration of various matching criteria**

Conflict Pairing Program (cont.)

Matching Possibilities

	CONFLICT OCCURS	CONFLICT DOES NOT OCCUR
ALERT	CP predicts conflict and it occurs (VA -- Valid Alert)	CP predicts conflict and it does not occur (FA -- False Alert)
NO ALERT	CP does not predict conflict and it occurs (MA -- Missed Alert)	CP does not predict conflict and It does not occur (NC -- correct No Call)
Total Number of Alerts	Total Number of Conflicts	Total Number of Non-Conflicts

Valid Alert in CP A, Missed Alert in CP B

		B			
		VA	MA	FA	NC
A	VA	SAME_VA	VA_MA	VA_FA	VA_NOMATCH
	MA	MA_VA	SAME_MA	MA_FA	MA_NOMATCH
	FA	FA_VA	FA_MA	SAME_FA	FA_NC
	NC	NOMATCH_VA	NOMATCH_MA	NC_FA	

Conflict Pairing Program (cont.)

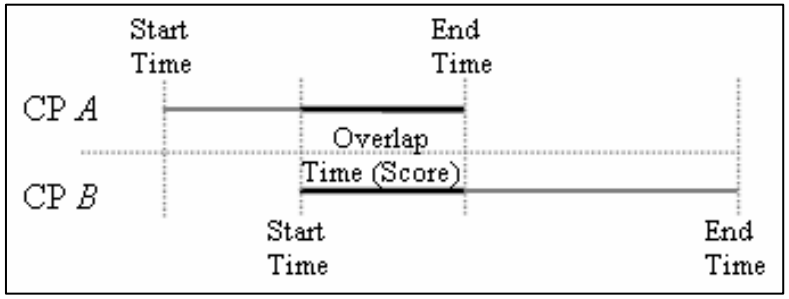
High-Level Abstraction

- **Same Ground Truth**
  - Both CPs have the same Radar Track Data
  - Actual Conflicts (VA, MA) in one CP must match with Actual Conflicts in the other
  - Actual Conflicts are split from False Alerts to create two matrices
- **Different Ground Truth**
  - Each CP has different Radar Track Data
  - Actual Conflicts can differ between CPs
  - One matrix for all alerts is used

Conflict Pairing Program (cont.)

Low-Level Abstraction: Matching Criteria

- **OOD allows exploration of new matching criteria (strategy)**
- **Most accurate strategy to date is Maximum Overlap Time**



Conflict Pairing Program (cont.)

Maximum Overlap Time Strategy

Min End Time - Max Start Time = Overlap Time

CP A		
#	Start	End
1	71200	71280
2	70000	70100
3	65000	68500
4	72400	72550

CP B		
#	Start	End
1	64980	68480
2	70100	70140
3	73040	73100

		B		
		1	2	3
A	1	-2720	-1060	-1760
	2	-1520	0	-2940
	3	3480	-1600	-4540
	4	-3920	-2260	-490

Example

Different Ground Truth: URET and ERAM

- **Results can be used to analyze differences**
  - Conflict Probing Algorithms
    - MA-VA pairs, FA-NC pairs
  - Radar Tracking Algorithms
    - NOMATCH pairs, FA-VA pairs

EVAL_CODE	COUNT
NOMATCH_VA	1
VA_NOMATCH	1
NOMATCH_MA	1
MA_VA	1
VA_MA	1
FA_VA	2
SAME_MA	3
VA_FA	5
NC_FA	142
FA_NC	169
SAME_VA	191
SAME_FA	341

This case will focus on NOMATCH_VA pair –

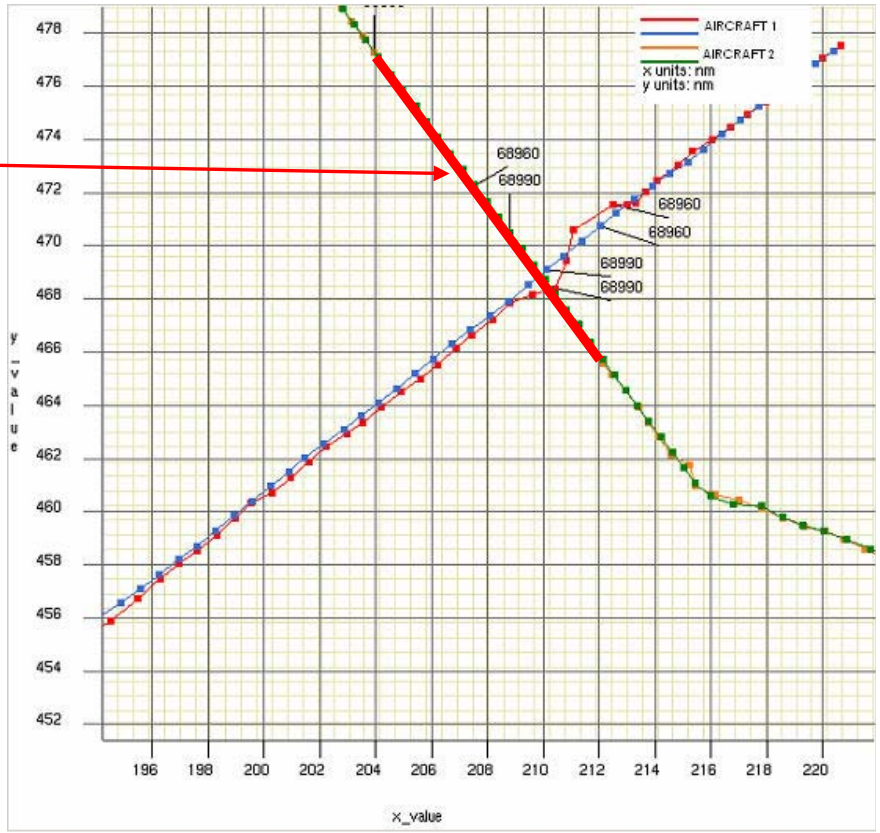
Radar Tracking difference

Example

Different Ground Truth: URET and ERAM (cont.)

Gap in URET

A Gap occurs when Radar Tracking has no data or illegitimate data



X, Y Plot

No Gap in ERAM

So, this is a Tracking algorithm difference

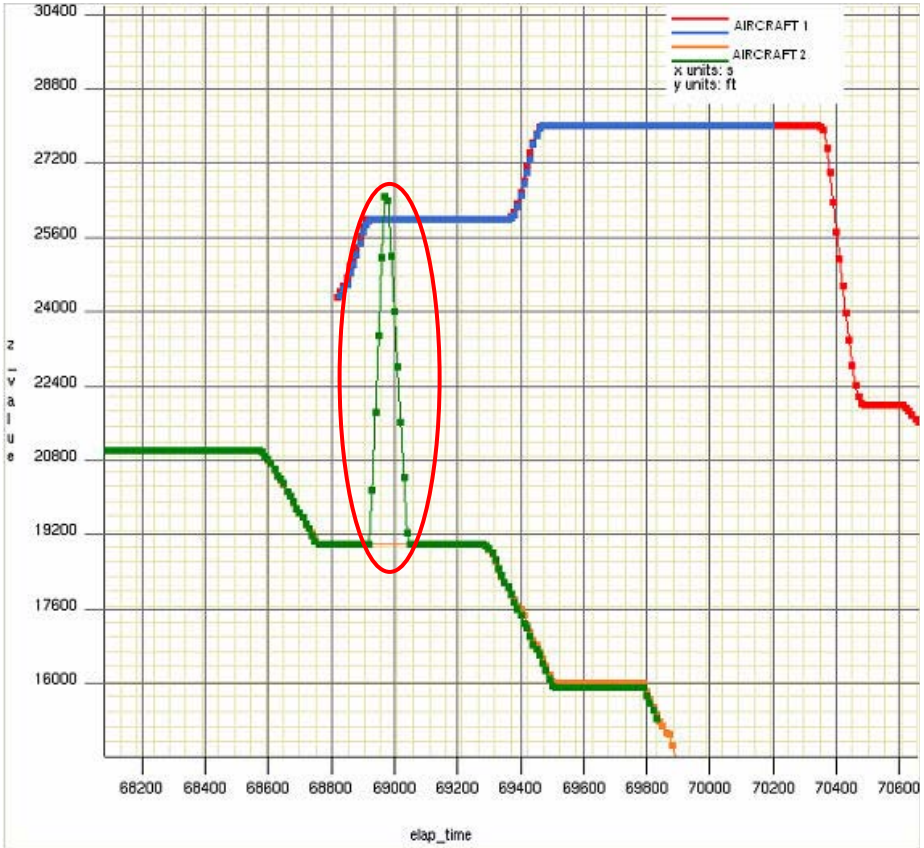
URET – Red, Orange

ERAM – Blue, Green

Example

Different Ground Truth: URET and ERAM (cont.)

Altitude spike in ERAM indicates bad Radar data



Each CP handled bad data differently

Time, Altitude Plot

Example

URET and Laboratory CP

- **Java-based simulation of URET**
- **Developed using original URET algorithms**
  - URET algorithms have changed
  - Performance of URET has increased considerably
- **Comparer currently being used to aid in updating Laboratory CP to closer simulate URET**

EVAL_CODE	COUNT
MA_VA	2
VA_MA	3
SAME_MA	3
DISCARD_VA	5
FA_DISCARD	151
DISCARD_FA	220
SAME_VA	234
FA_NC	335
SAME_FA	626
NC_FA	693
SAME_DISCARD	1400
DISCARD_NC	1409
NC_DISCARD	2387

Conclusions

- **Comparison Program has proven to be useful in development of several CPs**
- **Analysts can focus resources on analyzing results of comparison**
- **OOD allows research into alternate algorithms**

Future Work

- **Improve time efficiency of algorithm**
- **Research spatial matching criteria**