How Formal Methods Impels Discovery: A Short History of an Air Traffic Management Project

Ricky W. Butler
George Hagen
Jeffrey M. Maddalon
César A. Muñoz
Anthony Narkawicz
NASA Langley Research Center
Hampton, VA 23681, USA

Gilles Dowek
École polytechnique and INRIA
LIX, École polytechnique
91128 Palaiseau Cedex, France
web: http://shemesh.larc.nasa.gov/fm/

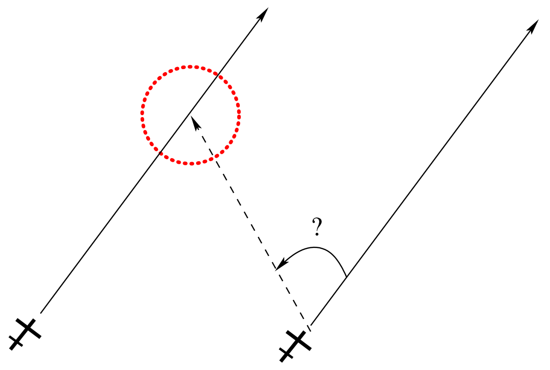
Introduction

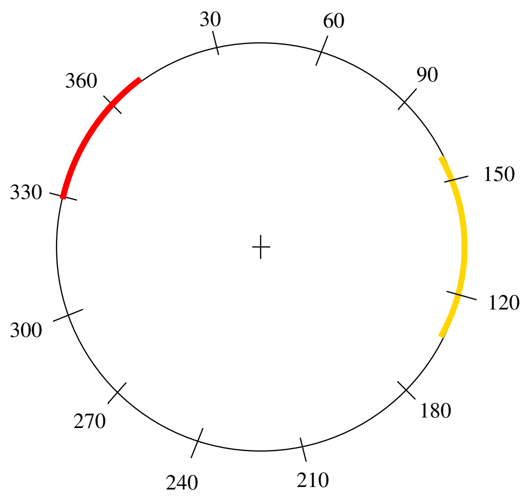
- Almost two years to complete and four additional formal methods researchers joined before we were done.
- A very interesting and enjoyable project:
  - The work resulted in a very elegant algorithm that is implemented in Java and C++,
  - The final algorithm was very different from our first ideas,
  - There were many, many discoveries that were surprising.
  - On the surface the problem looks simple, but looks can be deceiving and the problem is actually very subtle with many special cases.
- After a year, we developed three bands algorithms and published them in a NASA Technical Memorandum.
- We had formalized much of the mathematical development in PVS priot to publication but not all.
- Much to our surprise the subsequent final formal proof step found some deficiencies in our algorithms.
- Deficiencies were repaired – PVS proof Nov 2009.

Motivation For Prevention Bands

How does a pilot know that a maneuver will not cause a near-term conflict?

What are all the headings that will cause a conflict?



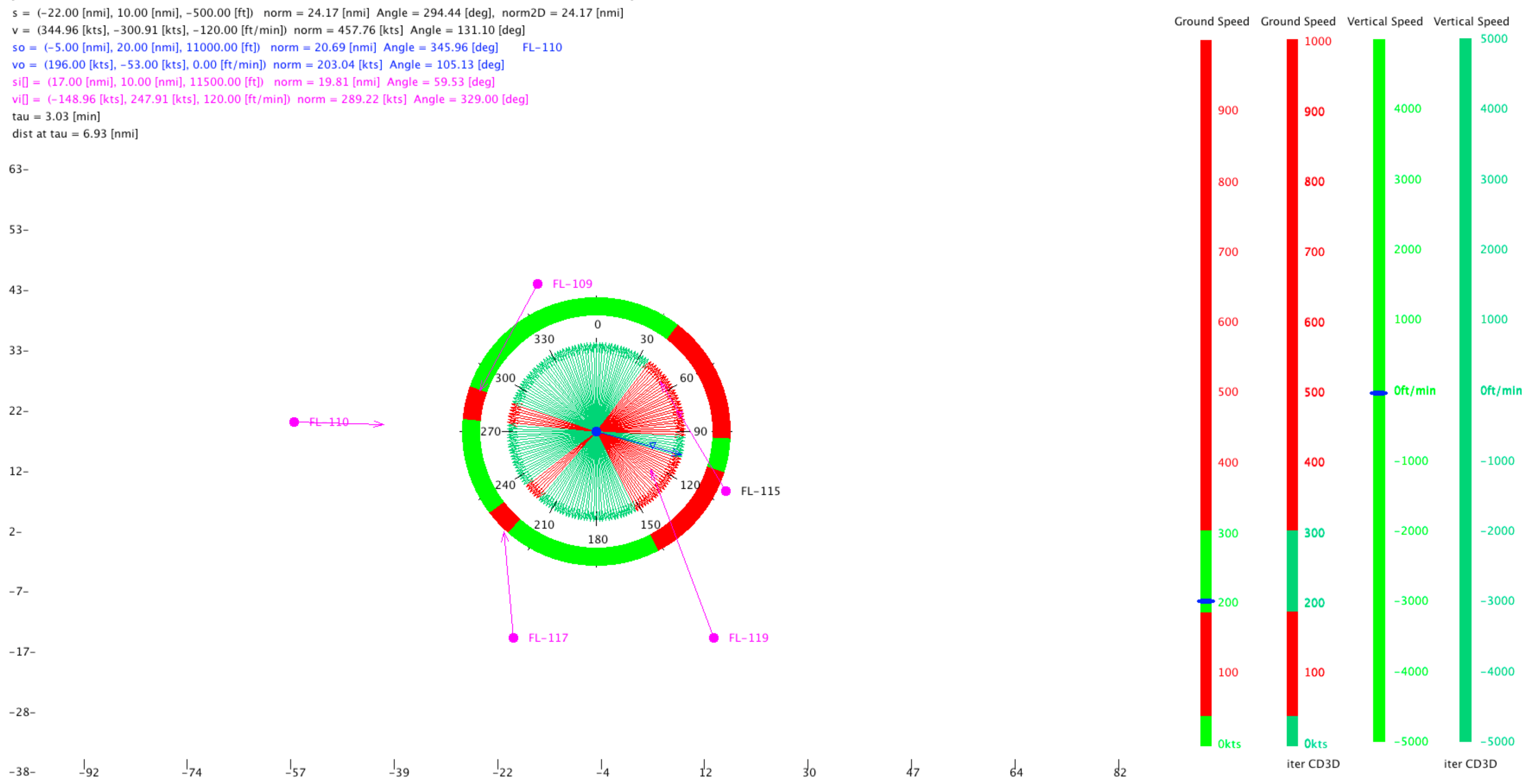


Prevention Bands show the pilot which maneuvers may cause

- **loss of separation within 5 minutes**
- **loss of separation within 3 minutes**

The Problem

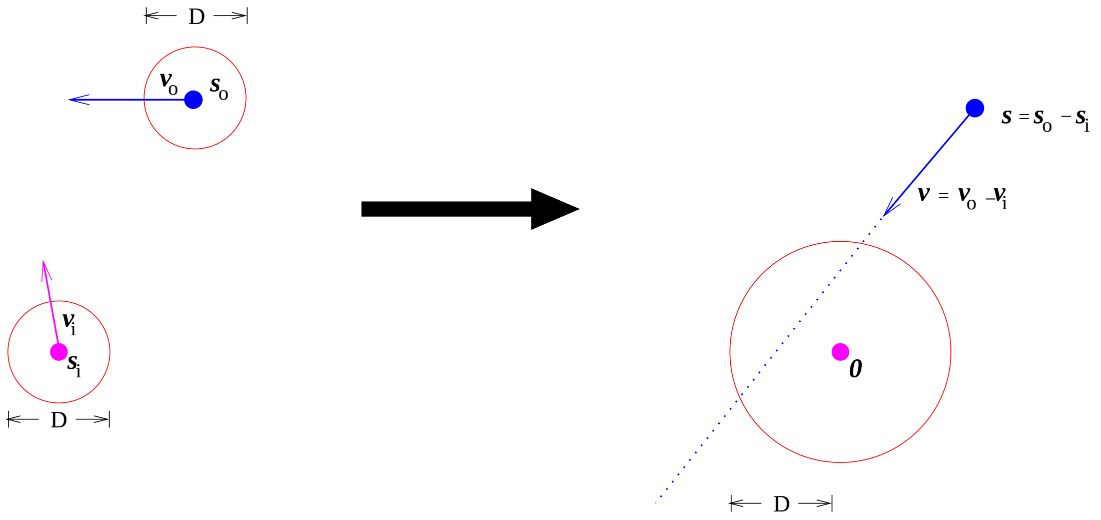
(1) Track-Angle (2) Ground-Speed (3) Vertical-Speed (Iterative & Analytic Solutions)



```
s =  (-22.00 [nmi], 10.00 [nmi], -500.00 [ft])   norm = 24.17 [nmi]  Angle = 294.44 [deg],  norm2D = 24.17 [nmi]
v =  (344.96 [kts], -300.91 [kts], -120.00 [ft/min])  norm = 457.76 [kts]  Angle = 131.10 [deg]
so =  (-5.00 [nmi], 20.00 [nmi], 11000.00 [ft])   norm = 20.69 [nmi]  Angle = 345.96 [deg]      FL-110
vo =  (196.00 [kts], -53.00 [kts], 0.00 [ft/min])  norm = 203.04 [kts]  Angle = 105.13 [deg]
si[] =  (17.00 [nmi], 10.00 [nmi], 11500.00 [ft])   norm = 19.81 [nmi]  Angle = 59.53 [deg]
vi[] =  (-148.96 [kts], 247.91 [kts], 120.00 [ft/min])  norm = 289.22 [kts]  Angle = 329.00 [deg]
tau = 3.03 [min]
dist at tau = 6.93 [nmi]
```

Notation

$\mathbf{s}_o$	3D vector	Initial position of the ownship aircraft
$\mathbf{v}_o$	3D vector	Initial velocity of the ownship aircraft
$\mathbf{s}_i$	3D vector	Initial position of the traffic aircraft
$\mathbf{v}_i$	3D vector	Initial velocity of the traffic aircraft



We use the relative frame with

$\mathbf{s} = \mathbf{s}_o - \mathbf{s}_i$.

$\mathbf{v} = \mathbf{v}_o - \mathbf{s}_i$.

Basic Definitions

A conflict there exists a future time t where the aircraft positions $\mathrm{s}_o + t\mathrm{v}_o$ and $\mathrm{s}_i + t\mathrm{v}_i$ are within a horizontal distance D of each other and where the aircraft are within vertical distance H of each other.

But it is convenient to decompose this into two predicates and express it in terms of the relative frame:

- A horizontal conflict occurs if there exists a future time t within the lookahead time T where the aircraft are within horizontal distance D of each other, i.e., $(s_x + tv_x)^2 + (s_y + tv_y)^2 < D^2$.

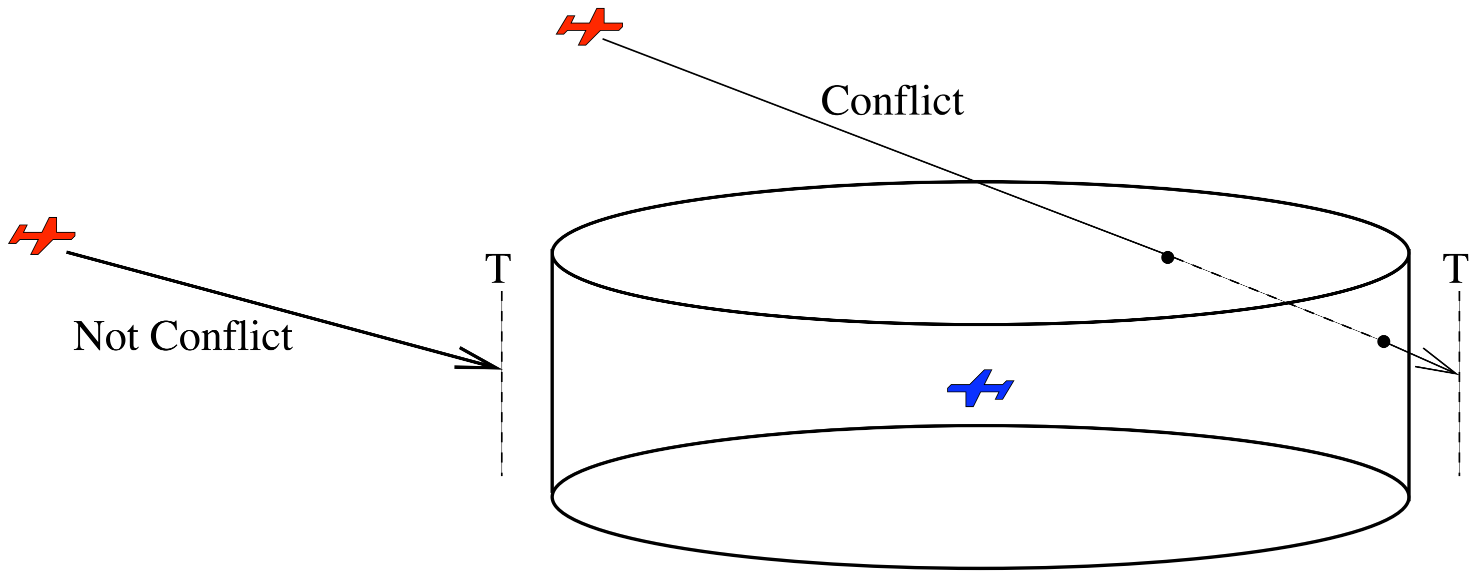
- A vertical conflict occurs if there exists a future time t within the lookahead time T where the aircraft are within horizontal distance H of each other, i.e., $|s_z + tv_z| < H$.

In the relative frame:

$$\begin{gathered}\texttt{conflict?}(\mathrm{s},\mathrm{v}) \equiv \exists\, 0 \le t \le T : \\ |\mathrm{s} + t\mathrm{v}|^2 < D^2 \text{ and } |s_z + tv_z| < H.\end{gathered} \tag{1}$$

Conflict With Lookahead Time

Exists a time t $\in [0, T]$ such that the red plane is inside the cylinder at time t.



Pairwise Method

We first recognized that each aircraft's contribution to the prevention band is independent of all other aircraft; thus, the problem neatly decomposes into two steps:

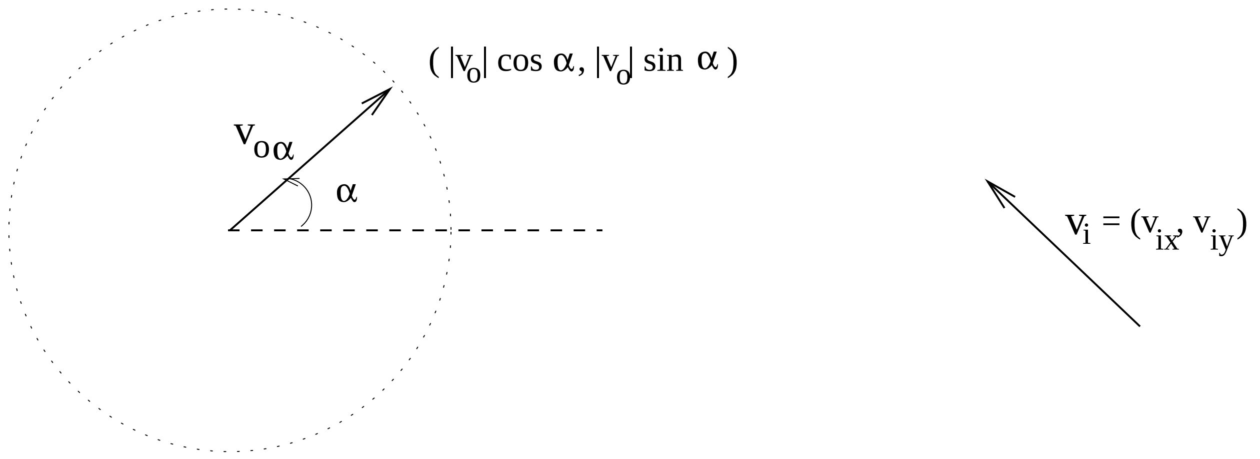
1. Solve the bands problem for the ownship relative to each other aircraft separately.
2. Merge all of the pairwise regions.

Iterative Solution

- **We quickly realized that an iterative solution was possible for the first step.**
- **We already had a formally proven, efficient algorithm available to us named `CD3D` that decides if a conflict occurs for specific values of s_o, v_o, s_i, and v_i, and parameters D, H, and T.**
- **Therefore, one need only to execute `CD3D` iteratively, varying the track angle from 0 to 360° per traffic aircraft.**

Search for an Analytical Solution

We begin with a non-translated perspective:



The track angle is denoted by α: For given vectors v_o and v_i, we need to find the track angles α such that the relative vector $\mathrm{v}_\alpha = \mathrm{v}_{o\alpha} - \mathrm{v}_i$:

$$\mathrm{v}_\alpha = (|\mathrm{v}_o| \cos \alpha - v_{ix},\ |\mathrm{v}_o| \sin \alpha - v_{iy}),$$

is not in conflict.

Initial Approach

- Divide the conflict prevention problem into simplifying cases.
- We decided to first solve the track bands problem in two dimensions without consideration of the lookahead time.
- The problem thus reduced to finding the tangent lines to the horizontal protection zone (in the relative frame of reference) as a function of α.
- We need solutions of $|\mathrm{s} + t\mathrm{v}_\alpha| = D$ or equivalently

$$(\mathrm{s} + t\mathrm{v}_\alpha)^2 = D^2 \tag{2}$$

- Expanding we obtain a quadratic equation $at^2 + bt + c = 0$ with $a = \mathrm{v}_\alpha{}^2$, $b = 2(\mathrm{s} \cdot \mathrm{v}_\alpha)$, and $c = \mathrm{s}^2 - D^2$.
- The tangent lines are precisely those where the discriminant of this equation is zero. In other words, where $b^2 - 4ac = 0$.

Initial Approach (cont)

But, expanding the dot products yield:

$$b^2 = 4[s_x(\omega \cos\alpha - v_{ix}) + s_y(\ \omega \sin\alpha - v_{iy})]^2$$
$$4ac = 4(\omega^2 - 2\omega(v_{ix}\cos\alpha + v_{iy}\sin\alpha) + v^2)(\mathrm{s}\cdot\mathrm{s} - D^2)$$

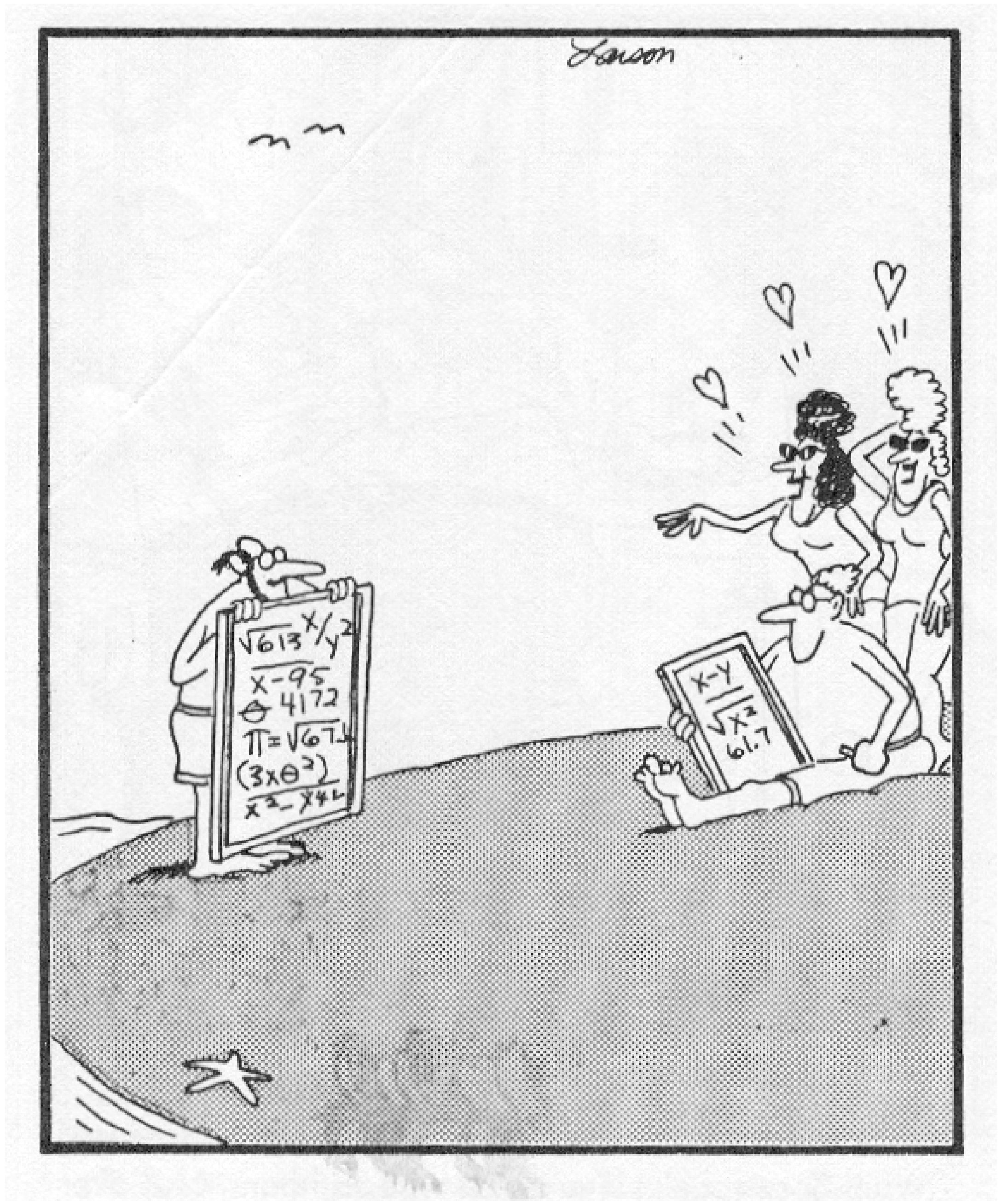
The discriminant finally expands into a complex second-order polynomial in $\sin\alpha$ and $\cos\alpha$.

But to solve for α, we need to eliminate the $\cos\alpha$ using the equation

$$\cos\alpha = \sqrt{1 - \sin^2\alpha}$$

The net result is an unbelievably complex fourth order polynomial in $\sin\alpha$:

$$4(sy\,viy\ .\ sx\,vix) - 4(sy\,viy\ .\ \sin(\alpha)\,\omega\,sy) - 4\left(sy\,viy\ .\ \sqrt{1-\sin(\alpha)^2}\,\omega\,sx\right) + 4(sy\,viy)^2 -$$
$$4\,c\,viy^2 + 8\sin(\alpha)\,c\,\omega\,viy + 4(sx\,vix\ .\ sy\,viy) - 4(sx\,vix\ .\ \sin(\alpha)\,\omega\,sy) - 4$$
$$\left(sx\,vix\ .\ \sqrt{1-\sin(\alpha)^2}\,\omega\,sx\right) + 4(sx\,vix)^2 - 4\,c\,vix^2 + 8\sqrt{1-\sin(\alpha)^2}\,c\,\omega\,vix - 4(\sin(\alpha)\,\omega\,sy\ .\ sy\,viy) -$$
$$4(\sin(\alpha)\,\omega\,sy\ .\ sx\,vix) + 4\left(\sin(\alpha)\,\omega\,sy\ .\ \sqrt{1-\sin(\alpha)^2}\,\omega\,sx\right) + 4(\sin(\alpha)\,\omega\,sy)^2 - 4$$
$$\left(\sqrt{1-\sin(\alpha)^2}\,\omega\,sx\ .\ sy\,viy\right) - 4\left(\sqrt{1-\sin(\alpha)^2}\,\omega\,sx\ .\ sx\,vix\right) + 4\left(\sqrt{1-\sin(\alpha)^2}\,\omega\,sx\ .\ \sin(\alpha)\,\omega\,sy\right) + 4$$
$$\left(\sqrt{1-\sin(\alpha)^2}\,\omega\,sx\right)^2 - 4\,c\,\omega^2 = 0$$



Sometimes, complex formulas are good.

Initial Approach (cont)

$$
\begin{aligned}
&4\,(sy\,viy \cdot sx\,vix) - 4\,(sy\,viy \cdot \sin(\alpha)\,\omega\,sy) - 4\left(sy\,viy \cdot \sqrt{1-\sin(\alpha)^2}\,\omega\,sx\right) + 4\,(sy\,viy)^2 - \\
&4\,c\,viy^2 + 8\sin(\alpha)\,c\,\omega\,viy + 4\,(sx\,vix \cdot sy\,viy) - 4\,(sx\,vix \cdot \sin(\alpha)\,\omega\,sy) - 4 \\
&\left(sx\,vix \cdot \sqrt{1-\sin(\alpha)^2}\,\omega\,sx\right) + 4\,(sx\,vix)^2 - 4\,c\,vix^2 + 8\sqrt{1-\sin(\alpha)^2}\,c\,\omega\,vix - 4\,(\sin(\alpha)\,\omega\,sy \cdot sy\,viy) - \\
&4\,(\sin(\alpha)\,\omega\,sy \cdot sx\,vix) + 4\left(\sin(\alpha)\,\omega\,sy \cdot \sqrt{1-\sin(\alpha)^2}\,\omega\,sx\right) + 4\,(\sin(\alpha)\,\omega\,sy)^2 - 4 \\
&\left(\sqrt{1-\sin(\alpha)^2}\,\omega\,sx \cdot sy\,viy\right) - 4\left(\sqrt{1-\sin(\alpha)^2}\,\omega\,sx \cdot sx\,vix\right) + 4\left(\sqrt{1-\sin(\alpha)^2}\,\omega\,sx \cdot \sin(\alpha)\,\omega\,sy\right) + 4 \\
&\left(\sqrt{1-\sin(\alpha)^2}\,\omega\,sx\right)^2 - 4\,c\,\omega^2 = 0
\end{aligned}
$$

- **Solving for α analytically would require the use of the quartic formulas.**
- **Although these formulas are complicated, such a program could probably be written in a day or two.**
- **But, how would we verify these solutions?**
- **After all, the quartic equations involve the use of complex analysis.**
- **Therefore, we began to look for simplifications.**

Approach 2

We remembered that a simplification of the discriminant that had been used in the design of the KB3D algorithm.

$$b^2 - 4ac = 0 \iff (\mathrm{s} \cdot \mathrm{v}) = R\,\epsilon\,\mathtt{det}(\mathrm{s}, \mathrm{v}) \tag{3}$$

where $\epsilon \in \{-1, +1\}$,

$\mathtt{det}(\mathrm{s}, \mathrm{v}) \equiv \mathrm{s}^\perp \cdot \mathrm{v}$,

$\mathrm{s}^\perp = (-s_y, s_x)$,

$R = \frac{\sqrt{\mathrm{s}^2 - D^2}}{D}$.

The beauty of the final form is that the equation is linear on v.

The two solutions are captured in the two values of ϵ.

When we instantiate v_α in this formula, we end up with a quadratic equation in $\mathtt{sin}\,\alpha$.

Approach 2 (cont)

Using this approach, we were able to derive the following solutions for α. If $\frac{|G|}{\sqrt{E^2+F^2}} \leq 1$ then in some 2π range, we have

$$\alpha_1 = \mathtt{asin}\left(\frac{G}{\sqrt{E^2+F^2}}\right) - \mathtt{atan}(E, F),$$
$$\alpha_2 = \pi - \mathtt{asin}\left(\frac{G}{\sqrt{E^2+F^2}}\right) - \mathtt{atan}(E, F),$$

where

$$E = \omega(R\epsilon s_x - s_y), \quad F = -\omega(R\epsilon s_y + s_x), \quad G = \mathrm{v}_i \cdot (R\epsilon\, \mathrm{s}^{\perp} - \mathrm{s}),$$

Since E, F, and G are all functions of ϵ, we have two pairs of α_1 and α_2 or a total of four total angles.

These angles are the places where the track prevention band changes color, assuming no lookahead time.

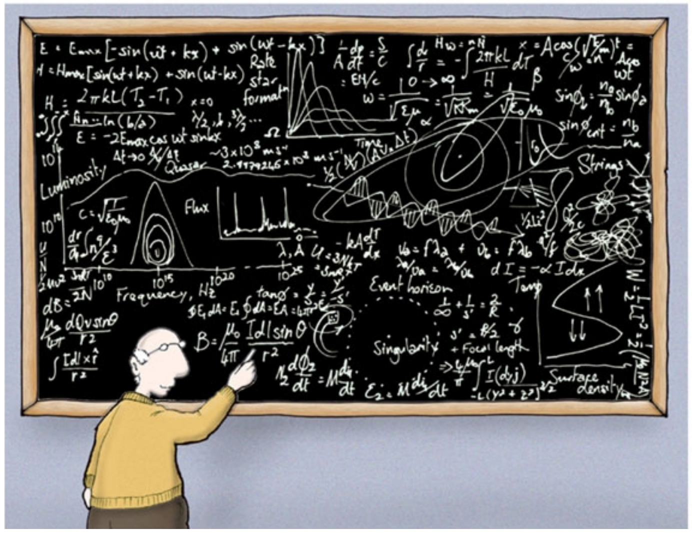
This result was formalized in the PVS theorem prover and implemented in Java.

Approach 2 (cont)

We were quite pleased with our initial geometric result and decided to present the result to our branch head and research director (June 2008).

During the presentation, Cesar Munoz said

> **“I think you can solve this problem without trigonometry!!”**



and he urged us to defer the use of trigonometry until the last possible moment.

In other words, he suggested that we solve for (v_α) without expanding its components.

We knew this was a good idea because so far we had only considered the 2-dimensional case with no lookahead time and even with these simplifications the trigonometry was killing us.

Approach 3

We soon realized that the 2D-problem was solvable by the KB3D resolutions called track lines, computed by the function `track_line`':

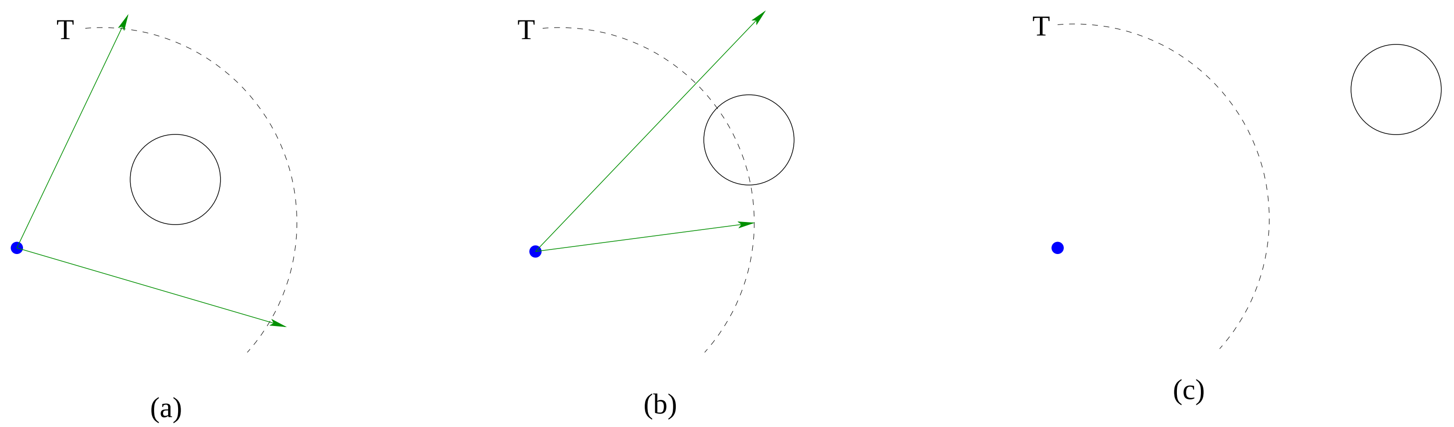
```
track_line(s,$\mathbf{v}_o$,$\mathbf{v}_i$,$\epsilon$,$\iota$) : Vect2 =
   LET $\mathbf{u}$ = tangent_line(s,$\epsilon$),
       a = $\mathbf{u}^2$,
       b = $\mathbf{u}\cdot\mathbf{v}_i$,
       c = $\mathbf{v\_i}^2$ - $\mathbf{v\_o}^2$ IN
   IF discriminant(a,2*b,c) $\geq$ 0 THEN
     LET k = root(a,2*b,c,$\iota$) IN
     IF k $\geq$ 0 THEN
       $\mathrm{k}\mathbf{u}+\mathbf{v}_i$
     ELSE
       $\mathbf{0}$
     ENDIF
   ELSE
     $\mathbf{0}$
   ENDIF
```

The function `track_line` returns the vector 0 when all track angles for the ownship yield a potential conflict.

Since ϵ and ι are ± 1, there are four possible track line solutions for given s, v_o, and v_i .

Lookahead Time

There are three distinct cases that appear when the lookahead time is considered: (a) the protection zone is totally within lookahead time, (b) the zone is partially within, and (c) the zone is totally beyond the lookahead time.



Cases (a) and (c) were easy to handle, but we realized that case (b) was going to be the hard one.

The key to solving track bands with a lookahead time is to find where the projected lookahead time intersects the protected zone.

Approach 3 (cont)

That is, plot where the relative position of the aircraft will be after T units of time in every possible direction given an unchanged ground speed and find the intersection points with the protection zone.

The function `track_circle`, also available in KB3D, provides these solutions, which are called track circle solutions.

```
track_circle(s,$\mathbf{v}_o$,$\mathbf{v}_i$,$t$,$\epsilon$,$\iota$): Vect2 =
    LET $\mathbf{w}$ = s - $t\mathbf{v}_i$,
        e = ($D^2$ - $\mathbf{s}^2$ - $t^2$(v_$o^2$-v_i$^2$))/2$t$ IN
    IF nz_vect2?($\mathbf{w}$) THEN
      LET $\mathbf{v}_o'$ = trk_only_dot($\mathbf{w}$,$\mathbf{v}_o$,$\mathbf{v}_i$,e,$\iota$) IN
      IF horizontal_dir_at?(s,$\mathbf{v}_o'$-$\mathbf{v}_i$,$t$,$\epsilon$) THEN
        $\mathbf{v}_o'$
      ELSE
        0
      ENDIF
   ELSE
      0
    ENDIF
```

We believe that the lookahead problem would not have been analytically tractable using the trigonometric approach pursued at first.

This switch to a pure algebraic approach was fundamental to achieving the final proof of the 3-dimensional bands algorithm.

The Track Bands Algorithm

We define a critical vector as a relative velocity vector where the color of the bands may change.

These critical vectors are

$$
\begin{aligned}
\mathrm{R}_{mm} &= \texttt{track\_line}(\mathrm{s}, \mathrm{v}_o, \mathrm{v}_i, -1, -1), \\
\mathrm{R}_{mp} &= \texttt{track\_line}(\mathrm{s}, \mathrm{v}_o, \mathrm{v}_i, -1, +1), \\
\mathrm{R}_{pm} &= \texttt{track\_line}(\mathrm{s}, \mathrm{v}_o, \mathrm{v}_i, +1, -1), \\
\mathrm{R}_{pp} &= \texttt{track\_line}(\mathrm{s}, \mathrm{v}_o, \mathrm{v}_i, +1, +1), \\
\mathrm{C}_{rm} &= \texttt{track\_circle}(\mathrm{s}, \mathrm{v}_o, \mathrm{v}_i, T_{red}, -1), \\
\mathrm{C}_{rp} &= \texttt{track\_circle}(\mathrm{s}, \mathrm{v}_o, \mathrm{v}_i, T_{red}, +1), \\
\mathrm{C}_{am} &= \texttt{track\_circle}(\mathrm{s}, \mathrm{v}_o, \mathrm{v}_i, T_{amber}, -1), \\
\mathrm{C}_{ap} &= \texttt{track\_circle}(\mathrm{s}, \mathrm{v}_o, \mathrm{v}_i, T_{amber}, +1), \\
\mathrm{C}_{em} &= \texttt{track\_circle}(\mathrm{s}, \mathrm{v}_o, \mathrm{v}_i, t_{entry}, -1), \\
\mathrm{C}_{ep} &= \texttt{track\_circle}(\mathrm{s}, \mathrm{v}_o, \mathrm{v}_i, t_{entry}, +1), \\
\mathrm{C}_{xm} &= \texttt{track\_circle}(\mathrm{s}, \mathrm{v}_o, \mathrm{v}_i, t_{exit}, -1), \\
\mathrm{C}_{xp} &= \texttt{track\_circle}(\mathrm{s}, \mathrm{v}_o, \mathrm{v}_i, t_{exit}, +1)
\end{aligned}
$$

Some of these vectors may be zero vectors in which case they are ignored.

Times t_{entry} and t_{exit} are the entry and exit times into the protection zone.

NOTE: There are analogous formulas for ground speed and vertical speed bands.

The Track Bands Algorithm (cont)

Calculate these vectors:

$$\mathrm{R}_{mm}, \mathrm{R}_{mp}, \mathrm{R}_{pm}, \mathrm{R}_{pp}, \mathrm{C}_{rm}, \mathrm{C}_{rp}, \mathrm{C}_{am}, \mathrm{C}_{ap}, \mathrm{C}_{em}, \mathrm{C}_{ep}, \mathrm{C}_{xm}, \mathrm{C}_{xp}$$

The corresponding track angles (using `atan`) are computed and sorted into a list of angles.

Next, the angles 0 and 2π are added to the list to provide appropriate bounding.

Then, a conflict probe (such as CD3D) is applied to an angle between each of the critical angles to determine which color the whole region should be painted: green, amber, or red).

This procedure is iterated between the ownship and all traffic aircraft.

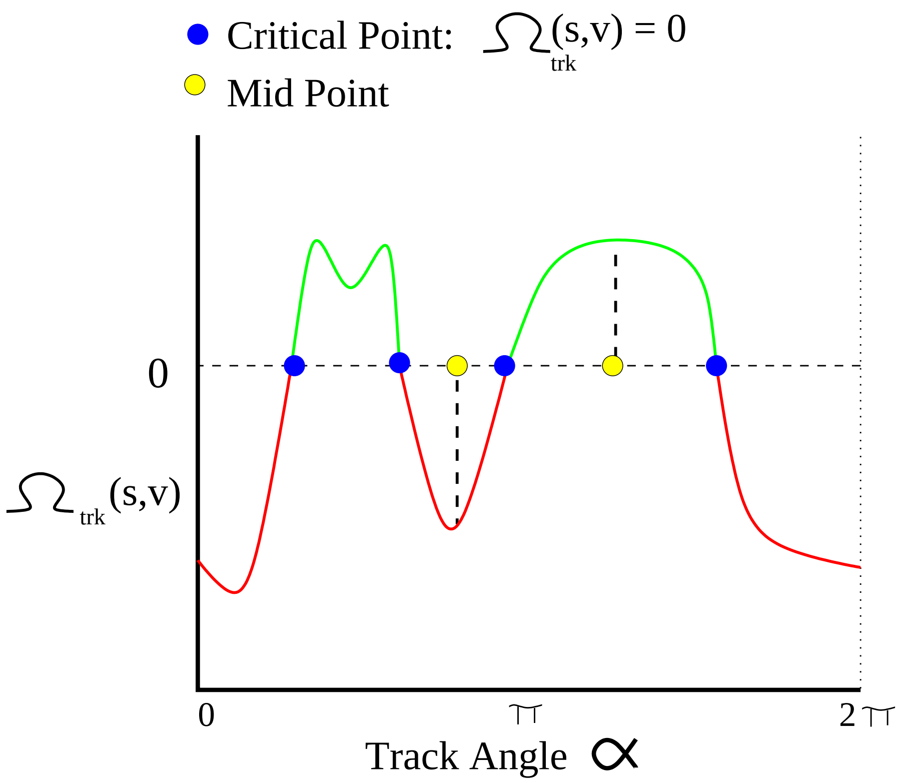
Finally, the resulting bands are merged.

Formal Verification of Pairwise Prevention Bands Algorithms

1. Find a function $\Omega_{\mathsf{trk}}(\mathrm{s}, \mathrm{v_o}, \mathrm{v_i}) : \mathbb{R} \longrightarrow \mathbb{R}$:

$$\Omega_{\mathsf{trk}}(\alpha) < 0 \iff \texttt{conflict?}(\mathrm{s}, \mathrm{v}_\alpha)$$

2. Prove that the critical vectors computed in the algorithm find all of the zeros of the function Ω_{trk},
3. Prove that the function Ω_{trk} is continuous.
4. Use the Intermediate Value theorem to deduce that any point in an open region, e.g. determines the color of the whole region.



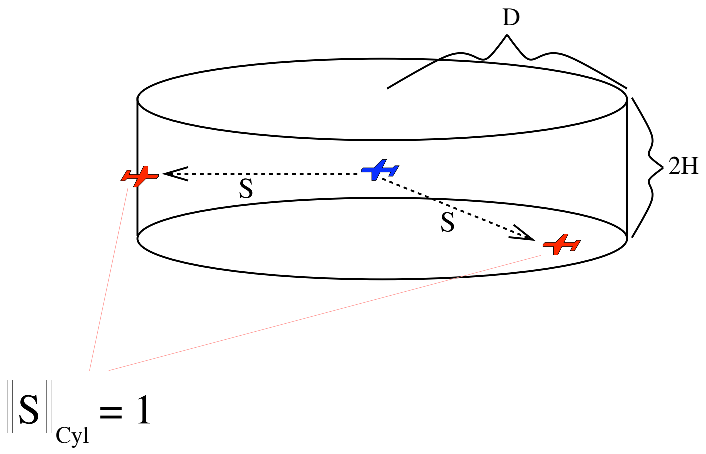
Formal Verification via Ω_{3D}

This concept is the notion of a normalized cylindrical length:

$$||\mathrm{u}||_{\mathsf{cyl}} = \max\left(\frac{\sqrt{u_x^2 + u_y^2}}{D}, \frac{|u_z|}{H}\right). \tag{4}$$

was the key to finding the needed Ω_{3D} function.

Cylindrical Length $||\boldsymbol{s}||_{cyl} = 1 \iff \boldsymbol{s}$ is "on the cylinder"

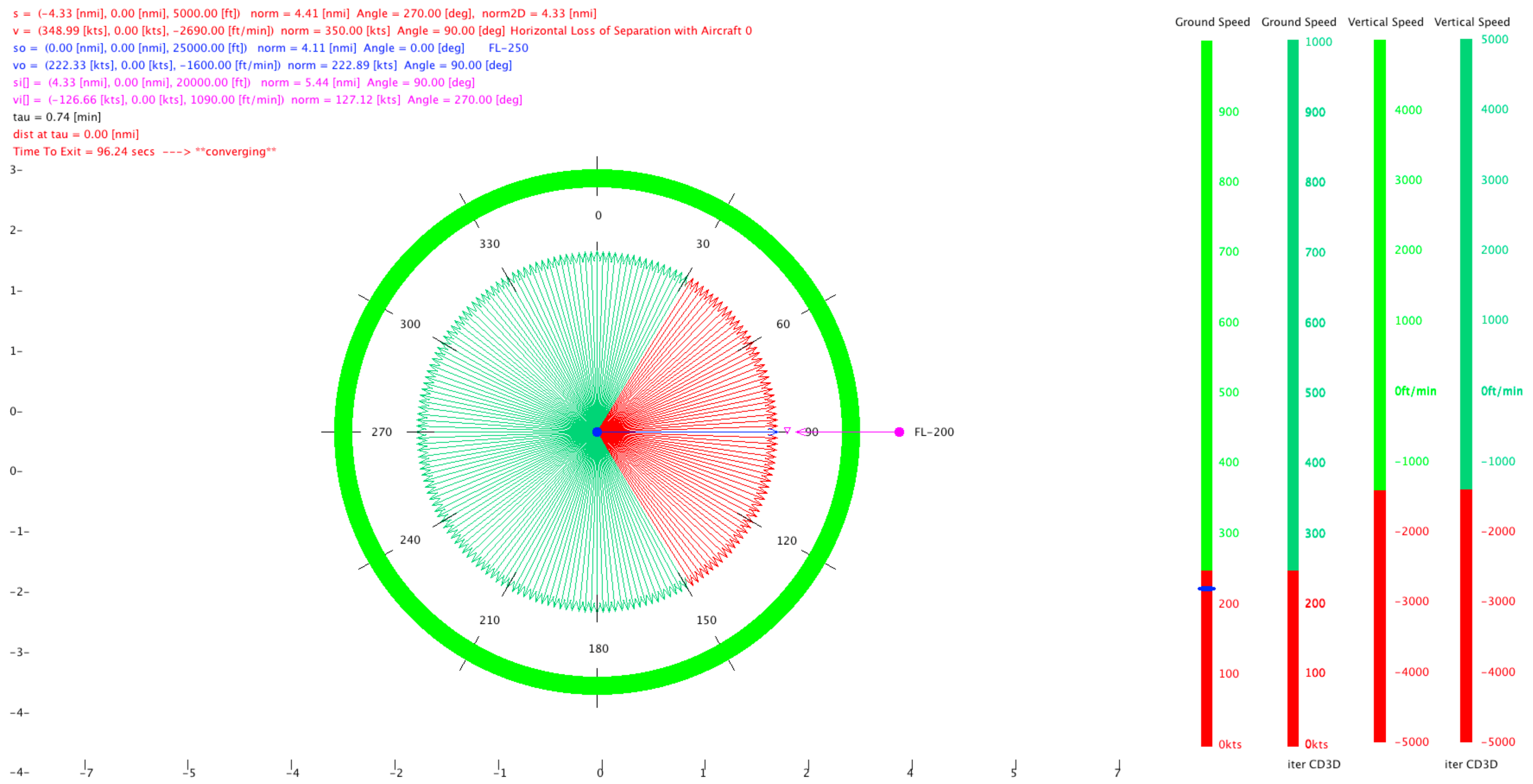


The Ω function can be defined as follows.

$$\Omega_{3D}(\mathrm{v}) = \min_{t\in[0,T]} ||\mathrm{s} + t\cdot\mathrm{v}||_{cyl} - 1, \tag{5}$$

Formal Verification (cont)

- **Verification Revealed Several Special Cases Where the Original 3-D Algorithm Was Incorrect**
- **<u>Verification</u>: 2-D: August 2009   3-D: Nov 2009**



```
s =  (-4.33 [nmi], 0.00 [nmi], 5000.00 [ft])   norm = 4.41 [nmi]  Angle = 270.00 [deg],  norm2D = 4.33 [nmi]
v =  (348.99 [kts], 0.00 [kts], -2690.00 [ft/min])  norm = 350.00 [kts]  Angle = 90.00 [deg] Horizontal Loss of Separation with Aircraft 0
so =  (0.00 [nmi], 0.00 [nmi], 25000.00 [ft])   norm = 4.11 [nmi]  Angle = 0.00 [deg]       FL-250
vo =  (222.33 [kts], 0.00 [kts], -1600.00 [ft/min])  norm = 222.89 [kts]  Angle = 90.00 [deg]
si[] =  (4.33 [nmi], 0.00 [nmi], 20000.00 [ft])   norm = 5.44 [nmi]  Angle = 90.00 [deg]
vi[] =  (-126.66 [kts], 0.00 [kts], 1090.00 [ft/min])  norm = 127.12 [kts]  Angle = 270.00 [deg]
tau = 0.74 [min]
dist at tau = 0.00 [nmi]
Time To Exit = 96.24 secs  ---> **converging**
```

The function Ω_{3D}

$$\begin{aligned}\Omega_{3D}(\boldsymbol{s},\boldsymbol{v}) &= \texttt{Minimum (Cylindrical) Dist Between Planes During [0,T]} \\ &= \min_{t\in[0,T]} ||\boldsymbol{s} + \boldsymbol{t}\cdot\boldsymbol{v}||_{cyl}\end{aligned}$$

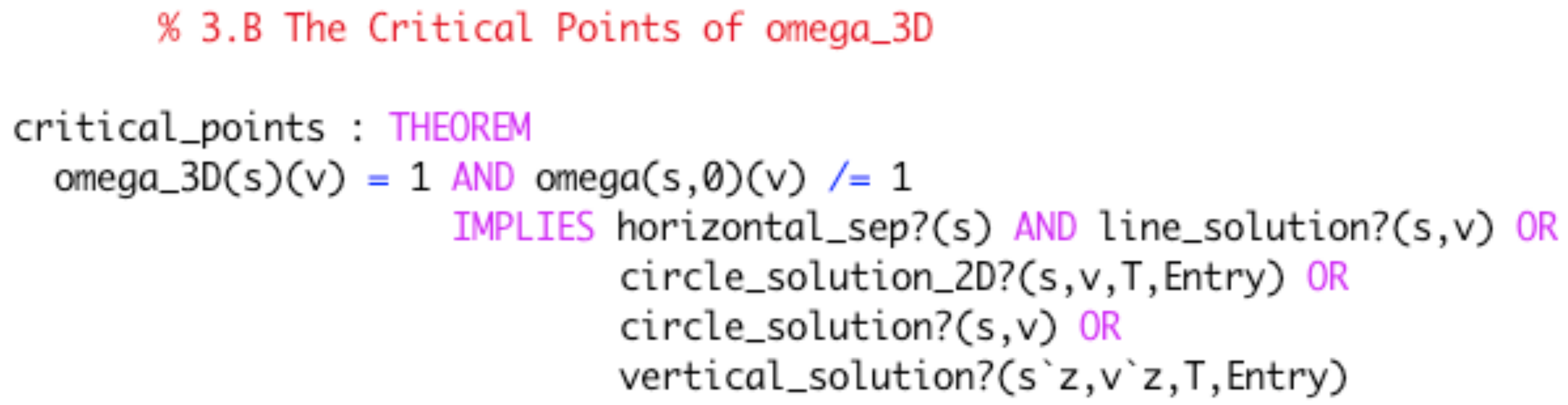
- Theorem: **Conflict** $\iff \Omega_{3D}(\boldsymbol{s},\boldsymbol{v}) < 1.$
- **Closed form is very messy**
- **Recently developed tools to formally verify that $\Omega_{3D}(\boldsymbol{s},\boldsymbol{v})$ is a continuous function of $\boldsymbol{v}$**
  - **the general proof of this fact requires the use of vector variant of the Heine-Cantor Theorem, i.e., if M is a compact metric space, then every continuous function $f : M \to N$, where N is a metric space, is uniformly continuous.**
  - **Furthermore, the minimum distance function may have flat areas.**
  - **Therefore, special attention has to be paid to the definition of Ω to guarantee that the set of critical points is finite.**
  - **Otherwise, it cannot be proven that the critical vector functions are complete.**

Verification of this algorithm:

Have we correctly classified all possible trajectories as conflict/not conflict?

- **$\Omega_{3D}(\boldsymbol{s}, \boldsymbol{v})$ is a continuous function of $\boldsymbol{v}$.**
- **Algorithm finds all critical points of Ω_{3D}.**

General theorem in PVS:

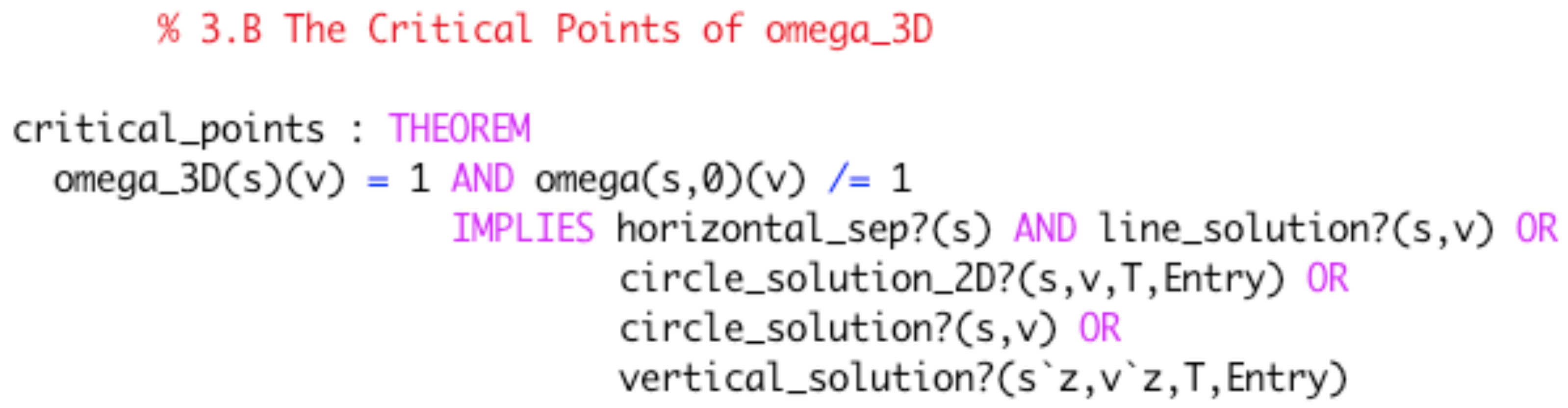


```
       % 3.B The Critical Points of omega_3D

 critical_points : THEOREM
   omega_3D(s)(v) = 1 AND omega(s,0)(v) /= 1
                      IMPLIES horizontal_sep?(s) AND line_solution?(s,v) OR
                              circle_solution_2D?(s,v,T,Entry) OR
                              circle_solution?(s,v) OR
                              vertical_solution?(s`z,v`z,T,Entry)
```

Verification of the $3 - D$ algorithm

- $\Omega_{3D}(\boldsymbol{s}, \boldsymbol{v})$ is a continuous function of $\boldsymbol{v}$. ✓(Aug/Sept 2009)
- Revised algorithm finds all critical points of Ω_{3D}. ✓ (Sept - Nov 2009)

General theorem in PVS:



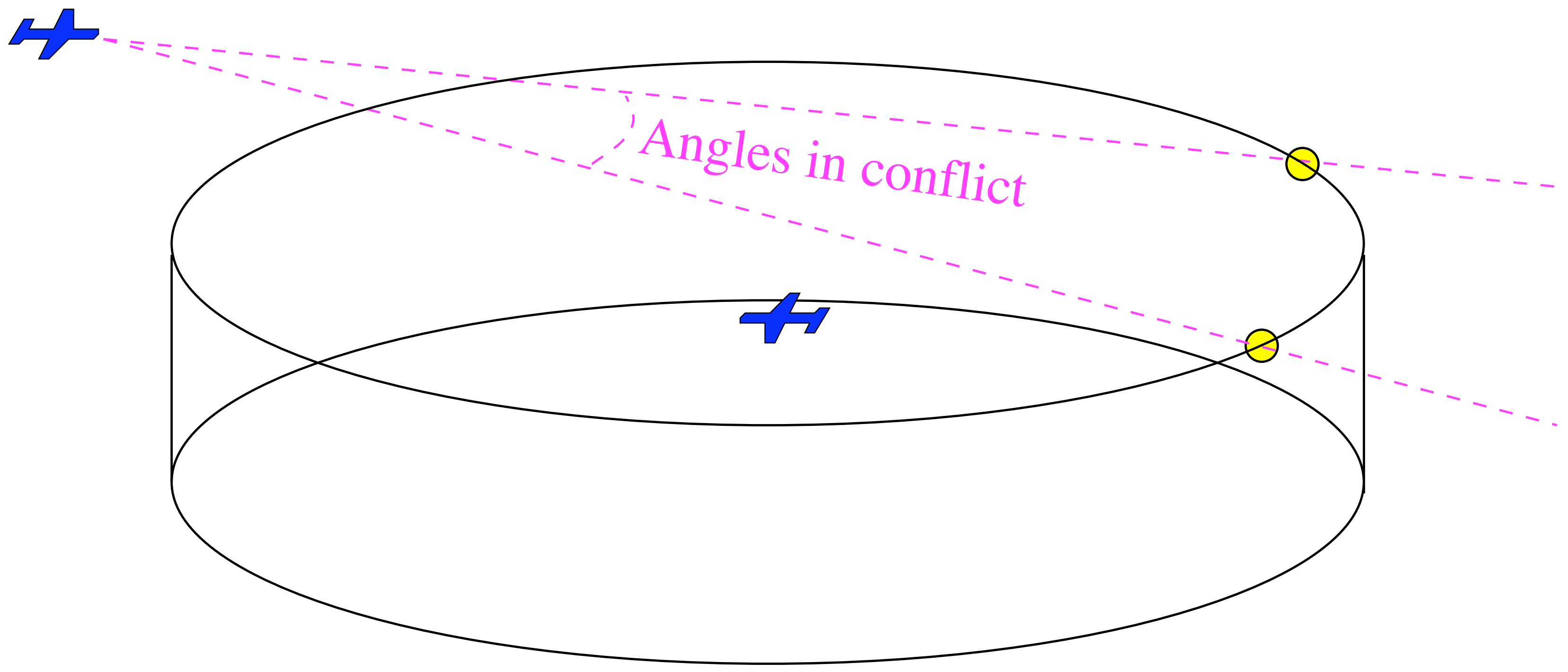
```
       % 3.B The Critical Points of omega_3D

critical_points : THEOREM
  omega_3D(s)(v) = 1 AND omega(s,0)(v) /= 1
                     IMPLIES horizontal_sep?(s) AND line_solution?(s,v) OR
                             circle_solution_2D?(s,v,T,Entry) OR
                             circle_solution?(s,v) OR
                             vertical_solution?(s`z,v`z,T,Entry)
```

Final theorem of algorithm correctness for ground-speed bands

```
%-------------------------------------------------%
% THEOREM: Ground Speed Green Bands Correctness   %
%-------------------------------------------------%

gs_green_band_3D : THEOREM
  FORALL (gso:(band)) :
     gs_band_3D?(sp,vo,vi)(band) IMPLIES
     (NOT cd3d?(sp,Vgs_3D(vo,vi)(gso)) IFF
      gs_green_3D?(sp,vo,vi)(band))

%------------------------------------------------%
% THEOREM: Ground Speed Red Bands Correctness    %
%------------------------------------------------%

gs_red_band_3D : THEOREM
  FORALL (gso:(band)) :
    gs_band_3D?(sp,vo,vi)(band) IMPLIES
    (cd3d?(sp,Vgs_3D(vo,vi)(gso)) IFF
      gs_red_3D?(sp,vo,vi)(band))
```

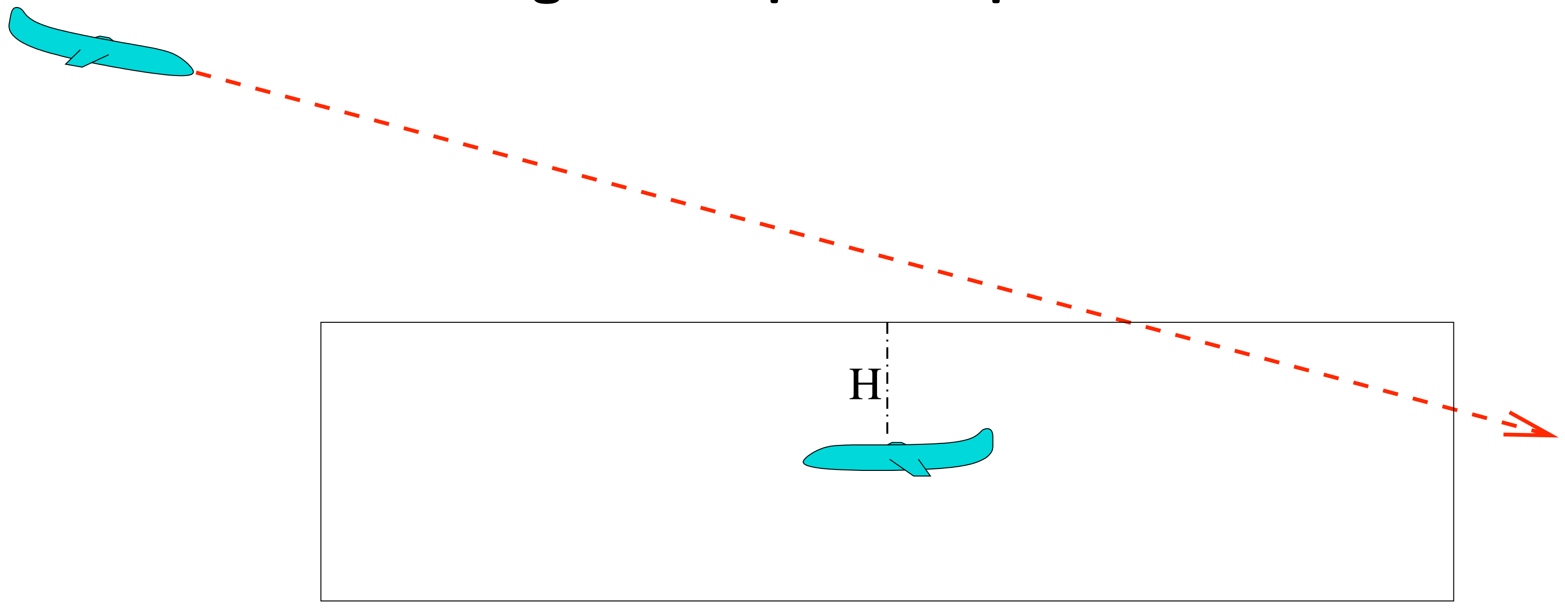
Missing solutions in the track-angle bands algorithm

Two missing track angle solutions to the equation $\Omega_{3D}(s, v) = 1$.



Missing solutions in the track-angle bands algorithm

Conflicts enter through the top of the protected zone



Results

- The proven functions are slightly different from the original ones presented in the NASA TM.
- For conflict prevention bands this is a safety issue, because a region that should be colored red can be colored green instead.
- Interestingly, those functions, which had been tested on over 10,000 test cases without any error manifestations, were in fact incorrect. The deficiencies in these functions were only found during the formal verification process!

Verification of the Merge Algorithm

Our original approach relied on complex reasoning about overlapping regions coupled with precedence rules: an amber region take precedence over a green region but not a red region.

We developed a Java version that “worked,” but it was soon obvious we needed to formally verify the merge algorithm.

The formal specification process enabled us to see two problems with our approach.

- Our algorithm was specialized to the precise problem we were working; almost any change to the system would require a new algorithm and therefore a new verification.
- Our algorithm was monolithic: there was no obvious decomposition into general pieces that could be verified once and used in different contexts.

To resolve these problems, we soon realized that standard set operations (set union, set difference, etc.) could be used to implement the merge.

Verification of the Merge Algorithm (cont.)

- By using set operations we had a well-defined specification of the key parts of our merge algorithm.
- However common implementations of sets in programming languages do not include efficient ways to deal with ranges of floating point numbers; therefore, we chose to implement our own.
- We then performed a code-level verification of the set union and set difference operations that were used in the merge algorithm.

Verification of the Merge Algorithm (cont.)

- The original Java implementation did not clearly indicate whether the endpoints of a band of green angles were part of that the green band, or if they were part of the next band (revealed by formal verification).
- It was not possible to exclusively use closed or open intervals for both union and difference operations: the use of one necessitates the use of the other.
  - For instance, removing a closed interval, which includes the endpoints, leaves us with open intervals—everything up to, but not including, these end points. Also, removing two adjacent open intervals leads to a left over point between them.

We ended up with the following approach:

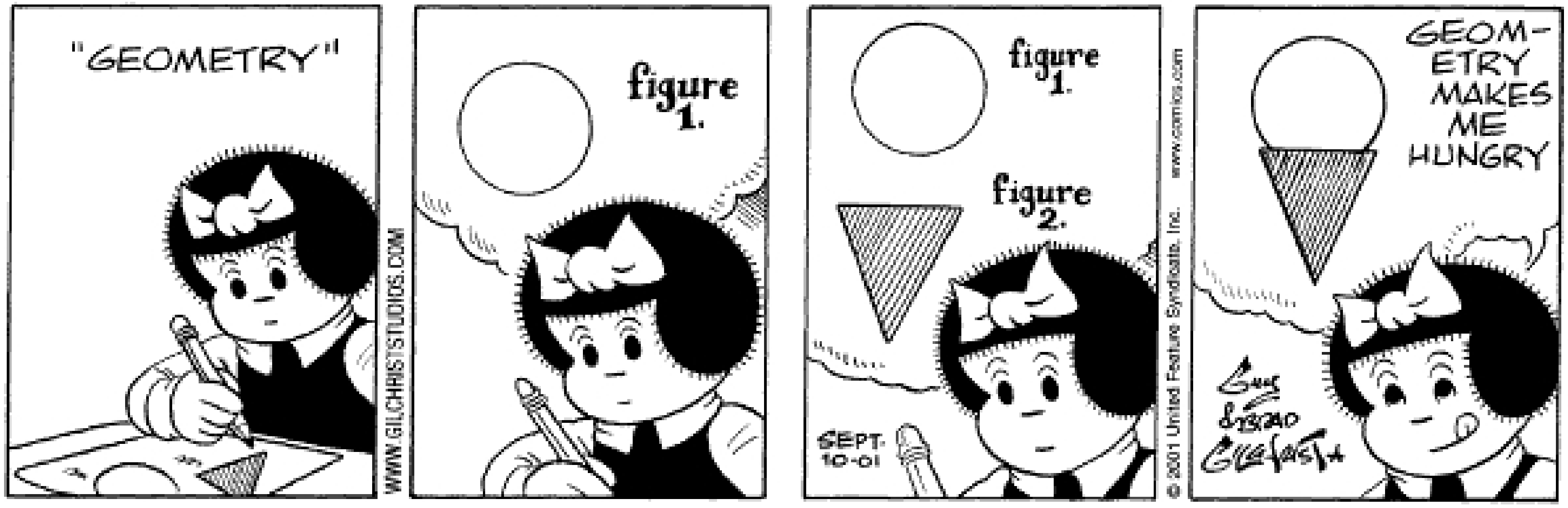
- The union operation would assume that inputs would be closed intervals and therefore, the result would be closed intervals.
- The difference operations would assume that the set to be subtracted would consist of only open intervals and therefore, the result would be only of closed intervals.

Verification of the Merge Algorithm (cont.)

- The formal verification of merge algorithm required us to think deeply about what the merge algorithm was trying to accomplish.
- During this analysis process we were able to develop an elegant solution which can be presented in a paragraph of text, instead of a complicated 400 line Java program with many special cases.
- In addition the formal verification process required us to clearly specify how our algorithm would behave at the points where there is a transition from one color to another.

Conclusions

- We have presented a short history of the development and formal verification of prevention bands algorithms.
- The resulting track-angle, ground speed, and vertical speed bands algorithms are far more simple than our earlier versions.
- The goal of completing a formal proof forced us to search for simplifications in the algorithms and in the underlying mathematical theories.
- A key insight that enabled the completion of this work is that trigonometric analysis should be deferred until the latest possible time.
- Although, the project took far longer than we expected, we are very pleased with the elegance and efficiencies of the discovered algorithms.


"GEOMETRY"
figure 1.
figure 1.
figure 2.
SEPT. 10-01
GEOMETRY MAKES ME HUNGRY
WWW.GILCHRISTSTUDIOS.COM
© 2001 United Feature Syndicate, Inc.
www.comics.com