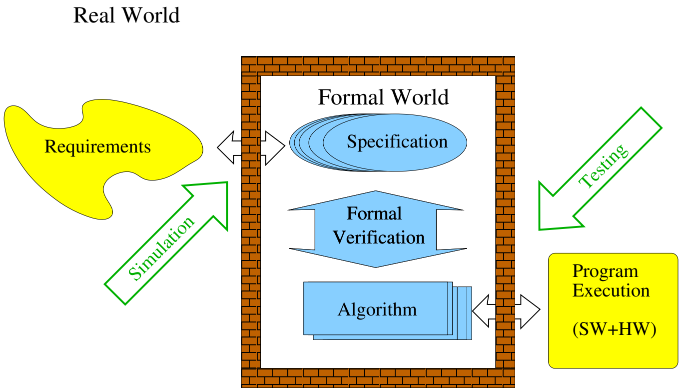
PVS in Practice

César A. Muñoz

NASA Langley Research Center
Cesar.A.Munoz@nasa.gov

PVS Tutorial 2017

The World According to PVS



All Models are Wrong

> *All models are wrong; the practical question is how wrong do they have to be to not be useful.*

G. Box and N. Draper, Empirical Model Building and Response Surfaces, 1987.

Prototype Verification System

- PVS (`http://pvs.csl.sri.com`) is developed by SRI International (`http://www.sri.com`).[1]
- Strongly typed specification language based on classical higher-order logic.
- Theorem prover with built-in decision procedures.

[1] Current version is 6.0.

Specification Language

- Classical logic: "To be or not to be" trivially holds!
- Higher-order logic: Quantification over sets and functions.
- Strongly typed language: All declarations have to be explicitly typed.
  - Predicate subtyping.
  - Dependent records.
  - Abstract Data Types.
  - Co-inductive types.

PVS's Type System is Undecidable

The following PVS specification

```
FermatNumber : TYPE =
  {n:above(2)| EXISTS (a,b,c:nat): a^n+b^n = c^n}

flt : FermatNumber
```

generates the following Type Correctness Condition (TCC)

```
FermatNumber_TCC1: OBLIGATION
 flt_TCC1: OBLIGATION EXISTS (x: FermatNumber): TRUE;
```

Mechanical Theorem Prover

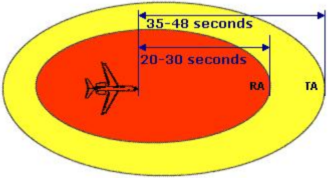
- Interactive and extensible theorem prover.
- Decision procedures for several theories: propositional logic, linear arithmetic, finite state machines, equality with uninterpreted functions, etc.
- Soundness-preserving strategy language.
- Semi-decision procedures for non-linear arithmetic.

PVS Standard Contributions

- NASA PVS Library: Large collection of PVS developments.
- ProofLite: Batch proving and proof scripting.
- PVSio: Animation and rapid prototyping.
- Manip and Field: Algebraic manipulation of real-valued expressions.
- Sturm and Tarski: Decision procedures for single-variable polynomials.
- Interval, Affine Arithmetic and Bernstein Polynomials: Semi-decision procedures for real-valued expressions.
- MetiTarski: External oracle for real-valued expressions.
- PRECiSA: Certifier of round-off floating point errors.
- Hypatheon: Database utility for PVS developments.

Traffic Alert and Collision Avoidance System (TCAS)*

- Family of of airborne systems designed to reduce the risk of mid-air collisions between *cooperative* aircraft.
- Mandated in the US for aircraft with greater than 30 seats or a maximum takeoff weight greater than 33,000 pounds.
- Current version, TCAS II V7.1, provides:
  - **Traffic Alerts** (TAs).
  - **(Vertical) Resolution Advisories** (RAs).



*Notional picture. Source of graphics: Wikipedia.

TCAS Alerting Logic

- TCAS logic uses 3-dimensional tracking of aircraft position.
- TCAS logic assumes linear trajectories that are extrapolation of current states.
- Parameters of extrapolated trajectories are compared to time and distance thresholds, whose values depend on **sensitivity level**.

TCAS Time and Distance TA Thresholds

Ownship Altitude (feet)	SL	TAU (sec)	DMOD (nmi)	ZTHR (feet)
Below 1000	2	20	0.30	850
1000 - 2350	3	25	0.33	850
2350 - 5000	4	30	0.48	850
5000 -10000	5	40	0.75	850
10000 - 20000	6	45	1.0	850
20000 - 42000	7	48	1.3	850
Above 42000	8	48	1.3	1200

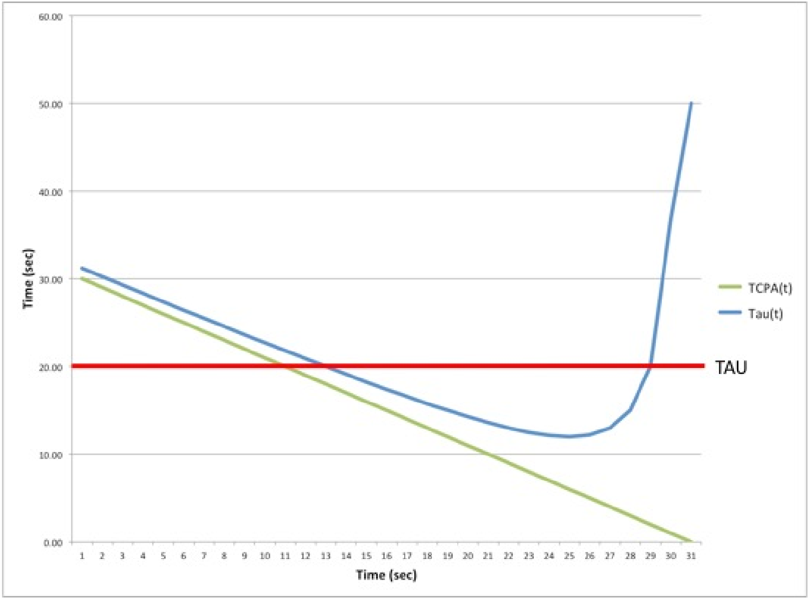
Range, Closure Rate, and τ

- Range (r): Horizontal distance between 2 aircraft.
- Closure rate ($-\dot{r}$): Negative of range rate.
- Tau (τ): Range over closure rate

$$\tau \equiv -\frac{r}{\dot{r}}.$$

- Tau's little secret: It is NOT the time of closest approach.

The Story of Tau (vs. TCPA)



Deconstructing TCAS 2D Core Alerting Logic

A Traffic Alert (TA) is issued when[1]

- The value of τ is below TAU threshold for appropriate sensitivity level or
- The distance r is below DMOD threshold for appropriate sensitivity level.

[1] This is a *simplified* version of the logic!

Aircraft State Information

Horizontal Plane

- $\mathbf{s}_o, \mathbf{v}_o$: Ownship's current position and velocity (2D Vectors).
- $\mathbf{s}_i, \mathbf{v}_i$: Intruder's current position and velocity (2D Vectors).

- $\mathbf{s}_o(t) = \mathbf{s}_o + t\mathbf{v}_o$: Ownship's position at time t.
- $\mathbf{s}_i(t) = \mathbf{s}_i + t\mathbf{v}_i$: Intruder's position at time t.

Reconstructing TCAS 2D Detection Logic

Assuming accurate vector information

Let $\mathbf{s} = \mathbf{s}_o - \mathbf{s}_i$ and $\mathbf{v} = \mathbf{v}_o - \mathbf{v}_i$,

$$
\begin{aligned}
r(\mathbf{s}) &\equiv \|\mathbf{s}\|, \\
\dot{r}(\mathbf{s}, \mathbf{v}) &\equiv \frac{\mathbf{s} \cdot \mathbf{v}}{\|\mathbf{s}\|}, \\
\tau(\mathbf{s}, \mathbf{v}) &\equiv -\frac{\|\mathbf{s}\|^2}{\mathbf{s} \cdot \mathbf{v}}, \\
t_{\text{cpa}}(\mathbf{s}, \mathbf{v}) &\equiv -\frac{\mathbf{s} \cdot \mathbf{v}}{\|\mathbf{v}\|^2} \\
\texttt{converging?}(\mathbf{s}, \mathbf{v}) &\equiv \mathbf{s} \cdot \mathbf{v} < 0, \\
\texttt{TCAS\_2D?}(\mathbf{s}, \mathbf{v}) &\equiv \begin{cases} \tau(\mathbf{s}, \mathbf{v}) < \texttt{TAU}_\ell & \text{if } \color{red}{\texttt{converging?}}(\mathbf{s}, \mathbf{v}), \\ r(\mathbf{s}) < \texttt{DMOD}_\ell & \text{otherwise.} \end{cases}
\end{aligned}
$$

($\texttt{TAU}_\ell$ and $\texttt{DMOD}_\ell$ are the thresholds for sensitivity level ℓ)

```
Units : THEORY
BEGIN

  ft   : MACRO posreal = 0.3048   % 1 foot in meters
  nmi  : MACRO posreal = 1852     % 1 nautical mile in meters
  min  : MACRO posreal = 60       % 1 minute in seconds
  hour : MACRO posreal = 3600     % 1 hour in seconds
  knt  : MACRO posreal = nmi/hour % 1 knot in m/s
  fpm  : MACRO posreal = ft/min   % 1 foot per minute in m/s

END Units
```

TCAS 2D Traffic Alerting Logic in PVS

A Simple Theory of Units

```
Units : THEORY
BEGIN

  ft   : MACRO posreal = 0.3048   % 1 foot in meters
  nmi  : MACRO posreal = 1852     % 1 nautical mile in meters
  min  : MACRO posreal = 60       % 1 minute in seconds
  hour : MACRO posreal = 3600     % 1 hour in seconds
  knt  : MACRO posreal = nmi/hour % 1 knot in m/s
  fpm  : MACRO posreal = ft/min   % 1 foot per minute in m/s

END Units
```

A Simple Theory of Units

```
Units : THEORY
BEGIN

  ft   : MACRO posreal = 0.3048   % 1 foot in meters
  nmi  : MACRO posreal = 1852     % 1 nautical mile in meters
  min  : MACRO posreal = 60       % 1 minute in seconds
  hour : MACRO posreal = 3600     % 1 hour in seconds
  knt  : MACRO posreal = nmi/hour % 1 knot in m/s
  fpm  : MACRO posreal = ft/min   % 1 foot per minute in m/s

END Units
```

TCAS Tables: Sensitivity Level

```
TCAS_tables : THEORY
BEGIN
  IMPORTING Units

  SensitivityLevel : TYPE = subrange(2,8)

  sensitivity_level(alt:nnreal) : SensitivityLevel =
    TABLE
    %+------------------------------------+---++
     |     0*ft <= alt AND alt <  1000*ft | 2 ||
    %+------------------------------------+---++
     |  1000*ft <= alt AND alt <  2350*ft | 3 ||
    %+------------------------------------+---++
     |  2350*ft <= alt AND alt <  5000*ft | 4 ||
    %+------------------------------------+---++
     |  5000*ft <= alt AND alt < 10000*ft | 5 ||
    %+------------------------------------+---++
     | 10000*ft <= alt AND alt < 20000*ft | 6 ||
    %+------------------------------------+---++
     | 20000*ft <= alt AND alt < 42000*ft | 7 ||
    %+------------------------------------+---++
     | ELSE                               | 8 ||
    %+------------------------------------+---++
    ENDTABLE
```

TCAS Tables: Sensitivity Level

```
TCAS_tables : THEORY
BEGIN
  IMPORTING Units

  SensitivityLevel : TYPE = subrange(2,8)

  sensitivity_level(alt:nnreal) : SensitivityLevel =
    TABLE
    %+-----------------------------------------+---++
     |     0*ft <= alt AND alt <  1000*ft | 2 ||
    %+-----------------------------------------+---++
     |  1000*ft <= alt AND alt <  2350*ft | 3 ||
    %+-----------------------------------------+---++
     |  2350*ft <= alt AND alt <  5000*ft | 4 ||
    %+-----------------------------------------+---++
     |  5000*ft <= alt AND alt < 10000*ft | 5 ||
    %+-----------------------------------------+---++
     | 10000*ft <= alt AND alt < 20000*ft | 6 ||
    %+-----------------------------------------+---++
     | 20000*ft <= alt AND alt < 42000*ft | 7 ||
    %+-----------------------------------------+---++
     | ELSE                               | 8 ||
    %+-----------------------------------------+---++
    ENDTABLE
```

TCAS Tables: Sensitivity Level

```
TCAS_tables : THEORY
BEGIN
  IMPORTING Units

  SensitivityLevel : TYPE = subrange(2,8)

  sensitivity_level(alt:nnreal) : SensitivityLevel =
    TABLE
    %+-----------------------------------+---++
     |     0*ft <= alt AND alt <  1000*ft | 2 ||
    %+-----------------------------------+---++
     |  1000*ft <= alt AND alt <  2350*ft | 3 ||
    %+-----------------------------------+---++
     |  2350*ft <= alt AND alt <  5000*ft | 4 ||
    %+-----------------------------------+---++
     |  5000*ft <= alt AND alt < 10000*ft | 5 ||
    %+-----------------------------------+---++
     | 10000*ft <= alt AND alt < 20000*ft | 6 ||
    %+-----------------------------------+---++
     | 20000*ft <= alt AND alt < 42000*ft | 7 ||
    %+-----------------------------------+---++
     | ELSE                               | 8 ||
    %+-----------------------------------+---++
    ENDTABLE
```

TCAS Tables: Sensitivity Level

```
TCAS_tables : THEORY
BEGIN
  IMPORTING Units

  SensitivityLevel : TYPE = subrange(2,8)

  sensitivity_level(alt:nnreal) : SensitivityLevel =
    TABLE
    %+-----------------------------------+---++
     |     0*ft <= alt AND alt <  1000*ft | 2 ||
    %+-----------------------------------+---++
     |  1000*ft <= alt AND alt <  2350*ft | 3 ||
    %+-----------------------------------+---++
     |  2350*ft <= alt AND alt <  5000*ft | 4 ||
    %+-----------------------------------+---++
     |  5000*ft <= alt AND alt < 10000*ft | 5 ||
    %+-----------------------------------+---++
     | 10000*ft <= alt AND alt < 20000*ft | 6 ||
    %+-----------------------------------+---++
     | 20000*ft <= alt AND alt < 42000*ft | 7 ||
    %+-----------------------------------+---++
     | ELSE                               | 8 ||
    %+-----------------------------------+---++
    ENDTABLE
```

TCAS Tables: Sensitivity Level

```
TCAS_tables : THEORY
BEGIN
  IMPORTING Units

  SensitivityLevel : TYPE = subrange(2,8)

  sensitivity_level(alt:nnreal) : SensitivityLevel =
    TABLE
    %+------------------------------------+---++
     |     0*ft <= alt AND alt <  1000*ft | 2 ||
    %+------------------------------------+---++
     |  1000*ft <= alt AND alt <  2350*ft | 3 ||
    %+------------------------------------+---++
     |  2350*ft <= alt AND alt <  5000*ft | 4 ||
    %+------------------------------------+---++
     |  5000*ft <= alt AND alt < 10000*ft | 5 ||
    %+------------------------------------+---++
     | 10000*ft <= alt AND alt < 20000*ft | 6 ||
    %+------------------------------------+---++
     | 20000*ft <= alt AND alt < 42000*ft | 7 ||
    %+------------------------------------+---++
     | ELSE                               | 8 ||
    %+------------------------------------+---++
    ENDTABLE
```

TCAS Tables: TAU, DMOD, and ZTHR

```
ThresholdSymbol : TYPE = { TAU, DMOD, ZTHR }

TA_thr(sl:SensitivityLevel,thr:ThresholdSymbol) : nnreal =
   TABLE sl ,     thr
       %--- +------+-----------+---------++
            |[ TAU |   DMOD    |   ZTHR  ]|
       %--- +------+-----------+---------++
        | 2 |   20 | 0.30*nmi |  850*ft ||
       %--- +------+-----------+---------++
        | 3 |   25 | 0.33*nmi |  850*ft ||
       %--- +------+-----------+---------++
        | 4 |   30 | 0.48*nmi |  850*ft ||
       %--- +------+-----------+---------++
        | 5 |   40 | 0.75*nmi |  850*ft ||
       %--- +------+-----------+---------++
        | 6 |   45 | 1.0*nmi  |  850*ft ||
       %--- +------+-----------+---------++
        | 7 |   48 | 1.3*nmi  |  850*ft ||
       %--- +------+-----------+---------++
        | 8 |   48 | 1.3*nmi  | 1200*ft ||
       %--- +------+-----------+---------++
   ENDTABLE
```

TCAS Tables: TAU, DMOD, and ZTHR

```
ThresholdSymbol : TYPE = { TAU, DMOD, ZTHR }

TA_thr(sl:SensitivityLevel,thr:ThresholdSymbol) : nnreal =
   TABLE sl ,     thr
       %--- +-------+-----------+----------++
            |[ TAU |   DMOD    |   ZTHR   ]|
       %--- +-------+-----------+----------++
        | 2 |   20 | 0.30*nmi |  850*ft ||
       %--- +-------+-----------+----------++
        | 3 |   25 | 0.33*nmi |  850*ft ||
       %--- +-------+-----------+----------++
        | 4 |   30 | 0.48*nmi |  850*ft ||
       %--- +-------+-----------+----------++
        | 5 |   40 | 0.75*nmi |  850*ft ||
       %--- +-------+-----------+----------++
        | 6 |   45 | 1.0*nmi  |  850*ft ||
       %--- +-------+-----------+----------++
        | 7 |   48 | 1.3*nmi  |  850*ft ||
       %--- +-------+-----------+----------++
        | 8 |   48 | 1.3*nmi  | 1200*ft ||
       %--- +-------+-----------+----------++
   ENDTABLE
```

TCAS Tables: TAU, DMOD, and ZTHR

```
ThresholdSymbol : TYPE = { TAU, DMOD, ZTHR }

TA_thr(sl:SensitivityLevel,thr:ThresholdSymbol) : nnreal =
   TABLE sl ,      thr
       %--- +------+----------+---------++
            |[ TAU |   DMOD   |   ZTHR  ]|
       %--- +------+----------+---------++
        | 2 |   20 | 0.30*nmi |  850*ft ||
       %--- +------+----------+---------++
        | 3 |   25 | 0.33*nmi |  850*ft ||
       %--- +------+----------+---------++
        | 4 |   30 | 0.48*nmi |  850*ft ||
       %--- +------+----------+---------++
        | 5 |   40 | 0.75*nmi |  850*ft ||
       %--- +------+----------+---------++
        | 6 |   45 | 1.0*nmi  |  850*ft ||
       %--- +------+----------+---------++
        | 7 |   48 | 1.3*nmi  |  850*ft ||
       %--- +------+----------+---------++
        | 8 |   48 | 1.3*nmi  | 1200*ft ||
       %--- +------+----------+---------++
   ENDTABLE
```

TCAS Tables: Type Correctness Conditions

```
% Disjointness TCC generated (at line 10, column 4) for
    % TABLE
    %   %-----------------------------------------------+---++
    %   | 0 * 0.3048 <= alt AND alt < 1000 * 0.3048      | 2 ||
    ...
    % ENDTABLE
  % proved - complete
sensitivity_level_TCC1: OBLIGATION
  FORALL (alt: nnreal):
         NOT ((0*0.3048 <= alt AND alt < 1000*0.3048) AND
               1000*0.3048 <= alt AND alt < 2350*0.3048) ...

% Coverage TCC generated (at line 34, column 5) for
    % TABLE sl, thr
    %      %+-----+-------------+---------------++
    %      |[ TAU | DMOD        | ZTHR          ]|
    ...
    % ENDTABLE
  % proved - complete
TA_thr_TCC1: OBLIGATION
  FORALL (sl: SensitivityLevel):
        sl=2 OR sl=3 OR sl=4 OR sl=5 OR sl=6 OR sl=7 OR sl=8;
```

TCAS Converging, Range, Closure Rate

```
TCAS_tau : THEORY
BEGIN

  %% All units are internal

  IMPORTING vectors@vectors_2D

  % s is a 2D relative position
  % v is a 2D relative velocity
  s,v : VAR Vect2

  converging?(s)(v) :bool =
    s*v < 0

  range(s) : nnreal = norm(s)

  closure_rate(s:Nz_vect2,v): real =
    -(s*v)/norm(s)
```

TCAS Converging, Range, Closure Rate

```
TCAS_tau : THEORY
BEGIN

  %% All units are internal

  IMPORTING vectors@vectors_2D

  % s is a 2D relative position
  % v is a 2D relative velocity
  s,v : VAR Vect2

  converging?(s)(v) :bool =
    s*v < 0

  range(s) : nnreal = norm(s)

  closure_rate(s:Nz_vect2,v): real =
    -(s*v)/norm(s)
```

TCAS Converging, Range, Closure Rate

```
TCAS_tau : THEORY
BEGIN

  %% All units are internal

  IMPORTING vectors@vectors_2D

  % s is a 2D relative position
  % v is a 2D relative velocity
  s,v : VAR Vect2

  converging?(s)(v) :bool =
    s*v < 0

  range(s) : nnreal = norm(s)

  closure_rate(s:Nz_vect2,v): real =
    -(s*v)/norm(s)
```

TCAS Tau

```
% Current tau is only defined when aircraft are converging
tau(s:Vect2,v:(converging?(s))) : nnreal =
  -sqv(s)/(s*v)

...

tau_TCC1: OBLIGATION FORALL (s: Vect2, v: (converging?(s))):
  (s*v) /= 0;

tau_TCC2: OBLIGATION
  FORALL (s: Vect2, v: (converging?(s))):
 -sqv(s)/(s*v) >= 0;
```

TCAS Tau

```
% Current tau is only defined when aircraft are converging
tau(s:Vect2,v:(converging?(s))) : nnreal =
  -sqv(s)/(s*v)

...
```

```
tau_TCC1: OBLIGATION FORALL (s: Vect2, v: (converging?(s))):
  (s*v) /= 0;

tau_TCC2: OBLIGATION
  FORALL (s: Vect2, v: (converging?(s))):
 -sqv(s)/(s*v) >= 0;
```

TCAS Tau

```
% Current tau is only defined when aircraft are converging
tau(s:Vect2,v:(converging?(s))) : nnreal =
  -sqv(s)/(s*v)

...
```

```
tau_TCC1: OBLIGATION FORALL (s: Vect2, v: (converging?(s))):
  (s*v) /= 0;

tau_TCC2: OBLIGATION
  FORALL (s: Vect2, v: (converging?(s))):
 -sqv(s)/(s*v) >= 0;
```

TCAS Tau

```
tau_def : LEMMA
  converging?(s)(v) IMPLIES
  tau(s,v) = range(s)/closure_rate(s,v)

%|- tau_def : PROOF
%|- (then
%|-  (skeep)
%|-  (expand* "range" "closure_rate" "tau" "converging?")
%|-  (grind-reals)
%|-  (rewrite "sq" :dir rl)
%|-  (rewrite "sq_norm"))
%|- QED
```

TCAS Tau

```
tau_def : LEMMA
  converging?(s)(v) IMPLIES
  tau(s,v) = range(s)/closure_rate(s,v)

%|- tau_def : PROOF
%|- (then
%|-  (skeep)
%|-  (expand* "range" "closure_rate" "tau" "converging?")
%|-  (grind-reals)
%|-  (rewrite "sq" :dir rl)
%|-  (rewrite "sq_norm"))
%|- QED
```

TCAS Tau

```
  tau_def : LEMMA
    converging?(s)(v) IMPLIES
    tau(s,v) = range(s)/closure_rate(s,v)

%|- tau_def : PROOF
%|- (then
%|-  (skeep)
%|-  (expand* "range" "closure_rate" "tau" "converging?")
%|-  (grind-reals)
%|-  (rewrite "sq" :dir rl)
%|-  (rewrite "sq_norm"))
%|- QED
```

TCPA vs. TCAS Tau

```
% Time of closest point of approach
tcpa(s,v) : nnreal =
  IF converging?(s)(v) THEN
    -(s*v)/sqv(v)
  ELSE
     0
  ENDIF

tau_ge_tcpa : LEMMA
  converging?(s)(v) IMPLIES
    tau(s,v) >= tcpa(s,v)
%|- tau_ge_tcpa : PROOF
%|- (then (grind) (metit *))
%|- QED
```

TCPA vs. TCAS Tau

```
  % Time of closest point of approach
  tcpa(s,v) : nnreal =
    IF converging?(s)(v) THEN
      -(s*v)/sqv(v)
    ELSE
       0
    ENDIF

  tau_ge_tcpa : LEMMA
    converging?(s)(v) IMPLIES
      tau(s,v) >= tcpa(s,v)
%|- tau_ge_tcpa : PROOF
%|- (then (grind) (metit *))
%|- QED
```

Generic TCAS 2D Alerting

```
TCAS_2D[(IMPORTING TCAS_tables) Thr : TCAS_Table] : THEORY
BEGIN

  IMPORTING TCAS_tau

  sl     : VAR SensitivityLevel
  so,si  : VAR Vect2     % Ownship's and intruder's positions
  vo,vi  : VAR Nz_vect2 % Ownship's and intruder's velocities

  TCAS_2D?(sl,so,vo,si,vi) : bool =
    LET s = so-si,
        v = vo-vi IN
      IF converging?(s)(v) THEN
        tau(s,v) < Thr(sl,TAU)
      ELSE
        range(s) < Thr(sl,DMOD)
      ENDIF
  ...
END TCAS_2D
```

Generic TCAS 2D Alerting

```
TCAS_2D[(IMPORTING TCAS_tables) Thr : TCAS_Table] : THEORY
BEGIN

  IMPORTING TCAS_tau

  sl     : VAR SensitivityLevel
  so,si  : VAR Vect2    % Ownship's and intruder's positions
  vo,vi  : VAR Nz_vect2 % Ownship's and intruder's velocities

  TCAS_2D?(sl,so,vo,si,vi) : bool =
    LET s = so-si,
        v = vo-vi IN
      IF converging?(s)(v) THEN
        tau(s,v) < Thr(sl,TAU)
      ELSE
        range(s) < Thr(sl,DMOD)
      ENDIF
  ...
END TCAS_2D
```

Generic TCAS 2D Alerting

```
TCAS_2D[(IMPORTING TCAS_tables) Thr : TCAS_Table] : THEORY
BEGIN

  IMPORTING TCAS_tau

  sl    : VAR SensitivityLevel
  so,si : VAR Vect2    % Ownship's and intruder's positions
  vo,vi : VAR Nz_vect2 % Ownship's and intruder's velocities

  TCAS_2D?(sl,so,vo,si,vi) : bool =
    LET s = so-si,
        v = vo-vi IN
      IF converging?(s)(v) THEN
        tau(s,v) < Thr(sl,TAU)
      ELSE
        range(s) < Thr(sl,DMOD)
      ENDIF
  ...
END TCAS_2D
```

Safety Property

```
TCAS_safe : CONJECTURE
  range(so-si) < Thr(sl,DMOD) IMPLIES
  TCAS_2D?(sl,so,vo,si,vi)
```

```
  |-------
{1}   FORALL (si: Vect2, sl: SensitivityLevel, so: Vect2,
                vi, vo: Nz_vect2):
          range(so - si) < Thr(sl, DMOD)
          IMPLIES TCAS_2D?(sl, so, vo, si, vi)

Rule? (skeep) (expand "TCAS_2D?") (ground)
{-1}  converging?(so - si)(vo - vi)
{-2}  range(so - si) < Thr(sl, DMOD)
  |-------
{1}   tau(so - si, vo - vi) < Thr(sl, TAU)

Rule? ...
```

Safety Property

```
  TCAS_safe : CONJECTURE
    range(so-si) < Thr(sl,DMOD) IMPLIES
    TCAS_2D?(sl,so,vo,si,vi)

  |-------
{1}   FORALL (si: Vect2, sl: SensitivityLevel, so: Vect2,
              vi, vo: Nz_vect2):
        range(so - si) < Thr(sl, DMOD)
        IMPLIES TCAS_2D?(sl, so, vo, si, vi)

Rule? (skeep) (expand "TCAS_2D?") (ground)
{-1}  converging?(so - si)(vo - vi)
{-2}  range(so - si) < Thr(sl, DMOD)
  |-------
{1}   tau(so - si, vo - vi) < Thr(sl, TAU)

Rule? ...
```

Safety Property

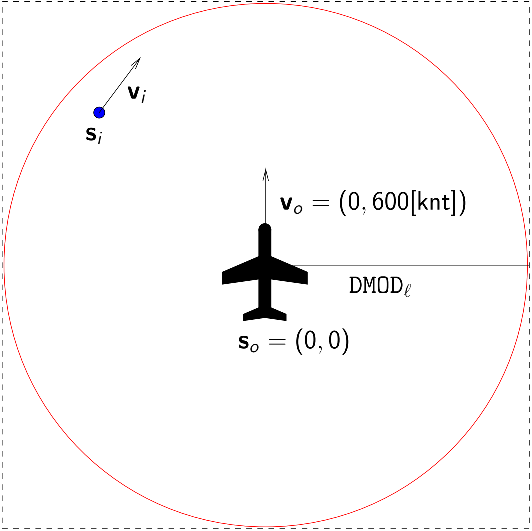
```
  TCAS_safe : CONJECTURE
    range(so-si) < Thr(sl,DMOD) IMPLIES
    TCAS_2D?(sl,so,vo,si,vi)

  |-------
{1}   FORALL (si: Vect2, sl: SensitivityLevel, so: Vect2,
                vi, vo: Nz_vect2):
        range(so - si) < Thr(sl, DMOD)
        IMPLIES TCAS_2D?(sl, so, vo, si, vi)

Rule? (skeep) (expand "TCAS_2D?") (ground)
{-1}  converging?(so - si)(vo - vi)
{-2}  range(so - si) < Thr(sl, DMOD)
  |-------
{1}   tau(so - si, vo - vi) < Thr(sl, TAU)

Rule? ...
```

Looking for a Counterexample to Safety Property



Finding Counterexample in PVS

```
TCAS_unsafe : THEORY
BEGIN

  IMPORTING TCAS_tables,TCAS_2D[TA_thr]
  sl : SensitivityLevel = 8
  so : Vect2 = (0,0)
  vo : Nz_vect2 = (0,600*knt)

  TCAS_unsafe : LEMMA EXISTS (six,siy,vix,viy: real):
      abs(six) <= TA_thr(sl,DMOD) AND
      abs(siy) <= TA_thr(sl,DMOD) AND
      abs(vix) <= 600*knt  AND abs(viy) <= 600*knt AND
      LET si : Vect2 = (six,siy), vi : Vect2 = (vix,viy) IN
      250*knt <= norm(vi) AND norm(vi) <= 600*knt AND
      converging?(so-si)(vo-vi) AND
      range(so-si) < TA_thr(sl,DMOD) AND
      tau(so-si,vo-vi) > TA_thr(sl,TAU)

END TCAS_unsafe
```

Finding Counterexample in PVS

```
TCAS_unsafe : THEORY
BEGIN

  IMPORTING TCAS_tables,TCAS_2D[TA_thr]
  sl : SensitivityLevel = 8
  so : Vect2 = (0,0)
  vo : Nz_vect2 = (0,600*knt)

  TCAS_unsafe : LEMMA EXISTS (six,siy,vix,viy: real):
      abs(six) <= TA_thr(sl,DMOD) AND
      abs(siy) <= TA_thr(sl,DMOD) AND
      abs(vix) <= 600*knt  AND abs(viy) <= 600*knt AND
      LET si : Vect2 = (six,siy), vi : Vect2 = (vix,viy) IN
      250*knt <= norm(vi) AND norm(vi) <= 600*knt AND
      converging?(so-si)(vo-vi) AND
      range(so-si) < TA_thr(sl,DMOD) AND
      tau(so-si,vo-vi) > TA_thr(sl,TAU)

END TCAS_unsafe
```

Finding Counterexample in PVS

```
TCAS_unsafe : THEORY
BEGIN

  IMPORTING TCAS_tables,TCAS_2D[TA_thr]
  sl : SensitivityLevel = 8
  so : Vect2 = (0,0)
  vo : Nz_vect2 = (0,600*knt)

  TCAS_unsafe : LEMMA EXISTS (six,siy,vix,viy: real):
      abs(six) <= TA_thr(sl,DMOD) AND
      abs(siy) <= TA_thr(sl,DMOD) AND
      abs(vix) <= 600*knt  AND abs(viy) <= 600*knt AND
      LET si : Vect2 = (six,siy), vi : Vect2 = (vix,viy) IN
      250*knt <= norm(vi) AND norm(vi) <= 600*knt AND
      converging?(so-si)(vo-vi) AND
      range(so-si) < TA_thr(sl,DMOD) AND
      tau(so-si,vo-vi) > TA_thr(sl,TAU)

END TCAS_unsafe
```

Finding Counterexample in PVS

```
  |-------
{1}   EXISTS (six, siy, vix, viy: real):
        ...

Rule? (grind :if-match nil :exclude "abs")

  |-------
{1}   EXISTS (six, siy, vix, viy: real):
              abs(six) <= 12038/5 AND abs(siy) <= 12038/5
          AND abs(vix) <= 926/3 AND abs(viy) <= 926/3
          AND 2315/18 <= sqrt(vix*vix + viy*viy)
          AND sqrt(vix*vix + viy*viy) <= 926/3
          AND six*vix + siy*viy - 926/3*siy < 0
          AND sqrt(six*six + siy*siy) < 12038/5
          AND -(six*six + siy*siy) /
               (six*vix + siy*viy - 926/3*siy) > 48
```

Finding Counterexample in PVS

```
  |-------
{1}   EXISTS (six, siy, vix, viy: real):
        ...

Rule? (grind :if-match nil :exclude "abs")

  |-------
{1}   EXISTS (six, siy, vix, viy: real):
              abs(six) <= 12038/5 AND abs(siy) <= 12038/5
          AND abs(vix) <= 926/3 AND abs(viy) <= 926/3
          AND 2315/18 <= sqrt(vix*vix + viy*viy)
          AND sqrt(vix*vix + viy*viy) <= 926/3
          AND six*vix + siy*viy - 926/3*siy < 0
          AND sqrt(six*six + siy*siy) < 12038/5
          AND -(six*six + siy*siy) /
               (six*vix + siy*viy - 926/3*siy) > 48
```

Finding Counterexample via Interval Arithmetic

```
IMPORTING interval_arith@strategies

  |-------
{1}   EXISTS (six, siy, vix, viy: real):
               abs(six) <= 12038/5 AND abs(siy) <= 12038/5
           AND abs(vix) <= 926/3 AND abs(viy) <= 926/3
           AND 2315/18 <= sqrt(vix*vix + viy*viy)
           AND sqrt(vix*vix + viy*viy) <= 926/3
           AND six*vix + siy*viy - 926/3*siy < 0
           AND sqrt(six*six + siy*siy) < 12038/5
           AND -(six*six + siy*siy) /
                (six*vix + siy*viy - 926/3*siy) > 48

Rule? (interval)
Q.E.D.
Run time  = 2.94 secs.
```

Finding Counterexample via Interval Arithmetic

```
IMPORTING interval_arith@strategies

  |-------
{1}   EXISTS (six, siy, vix, viy: real):
             abs(six) <= 12038/5 AND abs(siy) <= 12038/5
          AND abs(vix) <= 926/3 AND abs(viy) <= 926/3
          AND 2315/18 <= sqrt(vix*vix + viy*viy)
          AND sqrt(vix*vix + viy*viy) <= 926/3
          AND six*vix + siy*viy - 926/3*siy < 0
          AND sqrt(six*six + siy*siy) < 12038/5
          AND -(six*six + siy*siy) /
              (six*vix + siy*viy - 926/3*siy) > 48

Rule? (interval)
Q.E.D.
Run time  = 2.94 secs.
```

Finding Counterexample via Interval Arithmetic

```
IMPORTING interval_arith@strategies

  |-------
{1}   EXISTS (six, siy, vix, viy: real):
               abs(six) <= 12038/5 AND abs(siy) <= 12038/5
           AND abs(vix) <= 926/3 AND abs(viy) <= 926/3
           AND 2315/18 <= sqrt(vix*vix + viy*viy)
           AND sqrt(vix*vix + viy*viy) <= 926/3
           AND six*vix + siy*viy - 926/3*siy < 0
           AND sqrt(six*six + siy*siy) < 12038/5
           AND -(six*six + siy*siy) /
                (six*vix + siy*viy - 926/3*siy) > 48

Rule? (interval)
Q.E.D.
Run time  = 2.94 secs.
```

Finding Counterexample via Interval Arithmetic

```
  |-------
{1}   EXISTS (six, siy, vix, viy: real):
             abs(six) <= 12038/5 AND abs(siy) <= 12038/5
          AND ...

Rule? (interval :verbose? t)
----
Sequent holds for six = -54171/40, siy = 78247/40,
                  vix = -463/3, viy = 463/3
Splits: 42. Depth: 10
----
Q.E.D.
```

Finding Counterexample via Interval Arithmetic

```
  |-------
{1}   EXISTS (six, siy, vix, viy: real):
             abs(six) <= 12038/5 AND abs(siy) <= 12038/5
         AND ...

Rule? (interval :verbose? t)
----
Sequent holds for six = -54171/40, siy = 78247/40,
                  vix = -463/3, viy = 463/3
Splits: 42. Depth: 10
----
Q.E.D.
```

PVS Animation via PVSio

```
TCAS_unsafe : THEORY
BEGIN
  IMPORTING TCAS_tables, TCAS_2D[TA_thr]

  sl : SensitivityLevel = 8
  so : Vect2 = (0,0)
  vo : Nz_vect2 = (0,600*knt)

  ...

  si : Vect2 = (-54171/40,78247/40)
  vi : Nz_vect2 = (-463/3,463/3)

END TCAS_unsafe
```

M-x PVSio

```
<PVSio> si;
==>
(# x := -54171/40, y := 78247/40 #)

<PVSio> vi;
==>
(# x := -463/3, y := 463/3 #)

<PVSio> print(vi`x);
-154.33333

<PVSio> print(vi`x/knt);
-300

<PVSio> printf("si = (~f,~f), vi = (~f,~f)",
               (si`x/nmi,si`y/nmi,vi`x/knt,vi`y/knt));
si = (-0.73125,1.05625), vi = (-300.0,300.0)
```

M-x PVSio

```
<PVSio> si;
==>
(# x := -54171/40, y := 78247/40 #)

<PVSio> vi;
==>
(# x := -463/3, y := 463/3 #)

<PVSio> print(vi`x);
-154.33333

<PVSio> print(vi`x/knt);
-300

<PVSio> printf("si = (~f,~f), vi = (~f,~f)",
                (si`x/nmi,si`y/nmi,vi`x/knt,vi`y/knt));
si = (-0.73125,1.05625), vi = (-300.0,300.0)
```

M-x PVSio

```
<PVSio> si;
==>
(# x := -54171/40, y := 78247/40 #)

<PVSio> vi;
==>
(# x := -463/3, y := 463/3 #)

<PVSio> print(vi`x);
-154.33333

<PVSio> print(vi`x/knt);
-300

<PVSio> printf("si = (~f,~f), vi = (~f,~f)",
                (si`x/nmi,si`y/nmi,vi`x/knt,vi`y/knt));
si = (-0.73125,1.05625), vi = (-300.0,300.0)
```

M-x PVSio

```
<PVSio> si;
==>
(# x := -54171/40, y := 78247/40 #)

<PVSio> vi;
==>
(# x := -463/3, y := 463/3 #)

<PVSio> print(vi`x);
-154.33333

<PVSio> print(vi`x/knt);
-300

<PVSio> printf("si = (~f,~f), vi = (~f,~f)",
               (si`x/nmi,si`y/nmi,vi`x/knt,vi`y/knt));
si = (-0.73125,1.05625), vi = (-300.0,300.0)
```

M-x PVSio

```
<PVSio> si;
==>
(# x := -54171/40, y := 78247/40 #)

<PVSio> vi;
==>
(# x := -463/3, y := 463/3 #)

<PVSio> print(vi‘x);
-154.33333

<PVSio> print(vi‘x/knt);
-300

<PVSio> printf("si = (~f,~f), vi = (~f,~f)",
               (si‘x/nmi,si‘y/nmi,vi‘x/knt,vi‘y/knt));
si = (-0.73125,1.05625), vi = (-300.0,300.0)
```

M-x PVSio

```
<PVSio> si;
==>
(# x := -54171/40, y := 78247/40 #)

<PVSio> vi;
==>
(# x := -463/3, y := 463/3 #)

<PVSio> print(vi‘x);
-154.33333

<PVSio> print(vi‘x/knt);
-300

<PVSio> printf("si = (~f,~f), vi = (~f,~f)",
                (si‘x/nmi,si‘y/nmi,vi‘x/knt,vi‘y/knt));
si = (-0.73125,1.05625), vi = (-300.0,300.0)
```

M-x PVSio

```
<PVSio> si;
==>
(# x := -54171/40, y := 78247/40 #)

<PVSio> vi;
==>
(# x := -463/3, y := 463/3 #)

<PVSio> print(vi`x);
-154.33333

<PVSio> print(vi`x/knt);
-300

<PVSio> printf("si = (~f,~f), vi = (~f,~f)",
                (si`x/nmi,si`y/nmi,vi`x/knt,vi`y/knt));
si = (-0.73125,1.05625), vi = (-300.0,300.0)
```

M-x PVSio

```
<PVSio> si;
==>
(# x := -54171/40, y := 78247/40 #)

<PVSio> vi;
==>
(# x := -463/3, y := 463/3 #)

<PVSio> print(vi`x);
-154.33333

<PVSio> print(vi`x/knt);
-300

<PVSio> printf("si = (~f,~f), vi = (~f,~f)",
                (si`x/nmi,si`y/nmi,vi`x/knt,vi`y/knt));
si = (-0.73125,1.05625), vi = (-300.0,300.0)
```

M-x PVSio

```
<PVSio> si;
==>
(# x := -54171/40, y := 78247/40 #)

<PVSio> vi;
==>
(# x := -463/3, y := 463/3 #)

<PVSio> print(vi`x);
-154.33333

<PVSio> print(vi`x/knt);
-300

<PVSio> printf("si = (~f,~f), vi = (~f,~f)",
               (si`x/nmi,si`y/nmi,vi`x/knt,vi`y/knt));
si = (-0.73125,1.05625), vi = (-300.0,300.0)
```

M-x PVSio

```
<PVSio> si;
==>
(# x := -54171/40, y := 78247/40 #)

<PVSio> vi;
==>
(# x := -463/3, y := 463/3 #)

<PVSio> print(vi`x);
-154.33333

<PVSio> print(vi`x/knt);
-300

<PVSio> printf("si = (~f,~f), vi = (~f,~f)",
               (si`x/nmi,si`y/nmi,vi`x/knt,vi`y/knt));
si = (-0.73125,1.05625), vi = (-300.0,300.0)
```

M-x PVSio

```
<PVSio> converging?(so-si)(vo-vi);
==>
TRUE

<PVSio> print("Tau: "+tau(so-si,vo-vi)+
              ", TAU: "+TA_thr(sl,TAU)+
              ", TCPA: "+tcpa(so-si,vo-vi));
Tau: 60.9375, TAU: 48, TCPA: 1.95

<PVSio> print(range(so-si));
2379.2187
```

M-x PVSio

```
<PVSio> converging?(so-si)(vo-vi);
==>
TRUE

<PVSio> print("Tau: "+tau(so-si,vo-vi)+
              ", TAU: "+TA_thr(sl,TAU)+
              ", TCPA: "+tcpa(so-si,vo-vi));
Tau: 60.9375, TAU: 48, TCPA: 1.95

<PVSio> print(range(so-si));
2379.2187
```

M-x PVSio

```
<PVSio> converging?(so-si)(vo-vi);
==>
TRUE

<PVSio> print("Tau: "+tau(so-si,vo-vi)+
              ", TAU: "+TA_thr(sl,TAU)+
              ", TCPA: "+tcpa(so-si,vo-vi));
Tau: 60.9375, TAU: 48, TCPA: 1.95

<PVSio> print(range(so-si));
2379.2187
```

M-x PVSio

```
<PVSio> converging?(so-si)(vo-vi);
==>
TRUE

<PVSio> print("Tau: "+tau(so-si,vo-vi)+
              ", TAU: "+TA_thr(sl,TAU)+
              ", TCPA: "+tcpa(so-si,vo-vi));
Tau: 60.9375, TAU: 48, TCPA: 1.95

<PVSio> print(range(so-si));
2379.2187
```

M-x PVSio

```
<PVSio> converging?(so-si)(vo-vi);
==>
TRUE

<PVSio> print("Tau: "+tau(so-si,vo-vi)+
              ", TAU: "+TA_thr(sl,TAU)+
              ", TCPA: "+tcpa(so-si,vo-vi));
Tau: 60.9375, TAU: 48, TCPA: 1.95

<PVSio> print(range(so-si));
2379.2187
```

M-x PVSio

```
<PVSio> converging?(so-si)(vo-vi);
==>
TRUE

<PVSio> print("Tau: "+tau(so-si,vo-vi)+
              ", TAU: "+TA_thr(sl,TAU)+
              ", TCPA: "+tcpa(so-si,vo-vi));
Tau: 60.9375, TAU: 48, TCPA: 1.95

<PVSio> print(range(so-si));
2379.2187
```

PVSio Animations

```
alerts : void =
  LET i = query_int("Up to time? ") IN
  printf("TAs up to: ~a~%",i) &
  FORALL(t:upto(i)):
  IF TCAS_2D?(sl,so+t*vo,vo,si+t*vi,vi) THEN
    print(t+" ")
  ELSE
    skip
  ENDIF
```

```
<PVSio> alerts;
Up to time?
60
TAs up to 60:
2 3 4
```

PVSio Animations

```
alerts : void =
  LET i = query_int("Up to time? ") IN
  printf("TAs up to: ~a~%",i) &
  FORALL(t:upto(i)):
  IF TCAS_2D?(sl,so+t*vo,vo,si+t*vi,vi) THEN
    print(t+" ")
  ELSE
    skip
  ENDIF
```

```
<PVSio> alerts;
Up to time?
60
TAs up to 60:
2 3 4
```

PVSio Animations

```
alerts : void =
  LET i = query_int("Up to time? ") IN
  printf("TAs up to: ~a~%",i) &
  FORALL(t:upto(i)):
  IF TCAS_2D?(sl,so+t*vo,vo,si+t*vi,vi) THEN
    print(t+" ")
  ELSE
    skip
  ENDIF
```

```
<PVSio> alerts;
Up to time?
60
TAs up to 60:
2 3 4
```

PVSio Animations

```
alerts : void =
  LET i = query_int("Up to time? ") IN
  printf("TAs up to: ~a~%",i) &
  FORALL(t:upto(i)):
  IF TCAS_2D?(sl,so+t*vo,vo,si+t*vi,vi) THEN
    print(t+" ")
  ELSE
    skip
  ENDIF
```

```
<PVSio> alerts;
Up to time?
60
TAs up to 60:
2 3 4
```

PVSio Animations

```
alerts : void =
  LET i = query_int("Up to time? ") IN
  printf("TAs up to: ~a~%",i) &
  FORALL(t:upto(i)):
  IF TCAS_2D?(sl,so+t*vo,vo,si+t*vi,vi) THEN
    print(t+" ")
  ELSE
    skip
  ENDIF
```

```
<PVSio> alerts;
Up to time?
60
TAs up to 60:
2 3 4
```

TCAS Safe Future Interval

```
  |-------
{1}  FORALL(t:real): t ## [| 5, 60 |] IMPLIES
      NOT TCAS_2D?(sl,so+t*vo,vo,si+t*vi,vi)

Rule? (skeep)(grind)
{-1}  5 <= t
{-2}  t <= 60
{-3}  sqrt(6122593009/1600 - 36228361/60*t + 214369/9*(t*t) +
            -54171/40*(-54171/40) + 2*(-463/3*(-54171/40)*t)
            + -463/3*(-463/3)*t*t)
       < 12038/5
  |-------
{1}   214369/9*t - 36228361/120 + -463/3*(-54171/40) +
       -463/3*(-463/3)*t
       < 00

Rule? (interval -3 1)
Q.E.D.
```

TCAS Safe Future Interval

```
  |-------
{1}  FORALL(t:real): t ## [| 5, 60 |] IMPLIES
      NOT TCAS_2D?(sl,so+t*vo,vo,si+t*vi,vi)

Rule? (skeep)(grind)
{-1}  5 <= t
{-2}  t <= 60
{-3}  sqrt(6122593009/1600 - 36228361/60*t + 214369/9*(t*t) +
            -54171/40*(-54171/40) + 2*(-463/3*(-54171/40)*t)
            + -463/3*(-463/3)*t*t)
       < 12038/5
  |-------
{1}   214369/9*t - 36228361/120 + -463/3*(-54171/40) +
       -463/3*(-463/3)*t
       < 00

Rule? (interval -3 1)
Q.E.D.
```

TCAS Safe Future Interval

```
  |-------
{1}  FORALL(t:real): t ## [| 5, 60 |] IMPLIES
      NOT TCAS_2D?(sl,so+t*vo,vo,si+t*vi,vi)

Rule? (skeep)(grind)
{-1}  5 <= t
{-2}  t <= 60
{-3}  sqrt(6122593009/1600 - 36228361/60*t + 214369/9*(t*t) +
            -54171/40*(-54171/40) + 2*(-463/3*(-54171/40)*t)
            + -463/3*(-463/3)*t*t)
       < 12038/5
  |-------
{1}   214369/9*t - 36228361/120 + -463/3*(-54171/40) +
       -463/3*(-463/3)*t
       < 00

Rule? (interval -3 1)
Q.E.D.
```

TCAS Safe Future Interval

```
  |-------
{1}  FORALL(t:real): t ## [| 5, 60 |] IMPLIES
      NOT TCAS_2D?(sl,so+t*vo,vo,si+t*vi,vi)

Rule? (skeep)(grind)
{-1}  5 <= t
{-2}  t <= 60
{-3}  sqrt(6122593009/1600 - 36228361/60*t + 214369/9*(t*t) +
            -54171/40*(-54171/40) + 2*(-463/3*(-54171/40)*t)
            + -463/3*(-463/3)*t*t)
       < 12038/5
  |-------
{1}   214369/9*t - 36228361/120 + -463/3*(-54171/40) +
       -463/3*(-463/3)*t
       < 00

Rule? (interval -3 1)
Q.E.D.
```

Finally

```
top : THEORY
BEGIN

  IMPORTING Units,       % Unit conversion functions
            TCAS_tables, % TCAS threshold tables
            TCAS_tau,    % Definition of tau and TCPA
            TCAS_2D,     % 2-D alerting logic
            TCAS_safety  % Safety properties

END top
```

```
$ proveit -a
Processing ./top.pvs. Writing output to file ./top.summary
 Proof summary for theory TCAS_2D
    TCAS_safe..............unfinished
    Theory totals: 1 formulas, 1 attempted, 0 succeeded
Grand Totals: 16 proofs, 16 attempted, 15 succeeded (13.30 s)
```

Finally

```
top : THEORY
BEGIN

  IMPORTING Units,         % Unit conversion functions
            TCAS_tables,   % TCAS threshold tables
            TCAS_tau,      % Definition of tau and TCPA
            TCAS_2D,       % 2-D alerting logic
            TCAS_safety    % Safety properties

END top

$ proveit -a
Processing ./top.pvs. Writing output to file ./top.summary
 Proof summary for theory TCAS_2D
    TCAS_safe..............unfinished
    Theory totals: 1 formulas, 1 attempted, 0 succeeded
Grand Totals: 16 proofs, 16 attempted, 15 succeeded (13.30 s)
```

Finally

```
top : THEORY
BEGIN

  IMPORTING Units,         % Unit conversion functions
            TCAS_tables, % TCAS threshold tables
            TCAS_tau,      % Definition of tau and TCPA
            TCAS_2D,       % 2-D alerting logic
            TCAS_safety   % Safety properties

END top

$ proveit -a
Processing ./top.pvs. Writing output to file ./top.summary
 Proof summary for theory TCAS_2D
    TCAS_safe..............unfinished
    Theory totals: 1 formulas, 1 attempted, 0 succeeded
Grand Totals: 16 proofs, 16 attempted, 15 succeeded (13.30 s)
```