(19) Europäisches Patentamt
European Patent Office
Office européen des brevets

(11) EP 2 101 306 A1

(12) EUROPEAN PATENT APPLICATION

(43) Date of publication:
16.09.2009 Bulletin 2009/38

(51) Int Cl.:
G08G 5/00 *(2006.01)* ***G06F 15/00*** *(2006.01)*

(21) Application number: **09158253.6**

(22) Date of filing: **13.12.2004**

(84) Designated Contracting States:
AT BE BG CH CY CZ DE DK EE ES FI FR GB GR HU IE IS IT LI LT LU MC NL PL PT RO SE SI SK TR

(30) Priority: **03.12.2004 CA 2489122**

(62) Document number(s) of the earlier application(s) in accordance with Art. 76 EPC:
04257729.6 / 1 667 087

(71) Applicant: **NAV Canada**
Ottawa, ON K1P 5L6 (CA)

(72) Inventors:
- **Burnett, Kevin**
  Ottawa Ontario K1V 1P5 (CA)
- **Scully, Gregg**
  Surrey British Columbia V4A 3L7 (CA)
- **Davis, Don**
  Calgary Alberta T3A 1S9 (CA)
- **Krause, James**
  Beaumont Alberta T4X 1C9 (CA)
- **Cooper, Ken**
  Kelowna British Columbia V1Y 9S5 (CA)
- **Musclow, Ralph**
  Ottawa Ontario K1Y 3T8 (CA)
- **Beasley, Patrick**
  Kanata Ontario K2K 2JP (CA)

(74) Representative: **Skone James, Robert Edmund**
Gill Jennings & Every LLP
Broadgate House
7 Eldon Street
London EC2M 7LH (GB)

Remarks:
This application was filed on 20-04-2009 as a divisional application to the application mentioned under INID code 62.

(54) Visual aircraft spacing system and method

(57) An air traffic display aid and method that uses parameters for defining a heading for an alpha approach and a beta approach. These can be intersecting or converging runways, dependent or independent parallel runways or airways in enroute configurations. A target reference point is determined in dependence upon the alpha and beta approaches. And an image reference point is determined in dependence upon at least one of the target reference point, a difference between the headings, a characteristic of the beta approach and an offset. Single target mirror ghosting, in-trail ghosting on demand, associated configurations and user selectable resynchronization spacing is also supported.

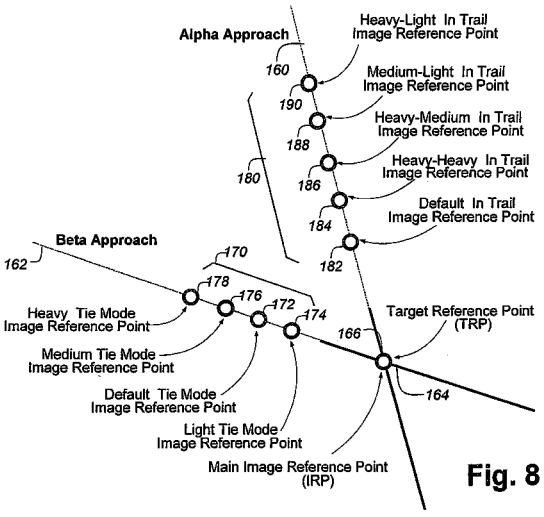


Fig. 8

EP 2 101 306 A1

Description

Field of the Invention

[0001] The present invention relates to air traffic display aids and is particularly concerned with tools for visual aircraft spacing.

Background of the Invention

[0002] Efficient use of airport runway systems is dependent upon air traffic controllers' ability to space aircraft for landing or take off as closely as allowed by air traffic regulations. This task is particularly challenging when multiple runways are involved, particularly in the case of converging runways or parallel runways that are close enough together for each to be dependent on the traffic on the other.

[0003] A solution to the converging runways scenario was taught by Anand D. Mundra, in U.S. Patent No. 4,890,232, issued on Dec. 26, 1989. The patent deals with the scenario of a first approach and a second approach intersecting the first. Mundra proposed displaying, along a line parallel to the second approach, a mirror image of aircraft on the first approach. The air traffic controller could then stagger the second approach aircraft by positioning them midway between the mirror images, thereby effectively spacing all approaching aircraft. As this was an early attempt to provide such an aid to air traffic controllers there were opportunities for improvement.

[0004] A paper "Converging Runway Display Aid As a Means to Increase Airport Capacity" presented in 2000, by Kevin Burnett, Patrick Beasley and Dr. Anand Mundra, discussed enhancements to Dr. Mundra's 1989 patent. The converging runway display aid (CRDA) described in the paper provides several different modes of displaying ghost images. As shown in Fig. 1, CDRA requires specification of an alpha approach 10 and a beta approach 12. Aircraft 14 have ghosts 16 projected from the alpha approach 10 to the beta approach 12. CRDA has two modes of display, so called "stagger mode" and "tie mode". The stagger mode is the same mode as taught in the patent, i.e. where the air traffic controller staggers the aircraft by having them move to a position between the ghost images. The tie mode is illustrated in Fig. 1. With the tie mode CDRA provides a tie mode spacing 18 between where the ghost image would appear if just mirrored and the actual placement of the ghost image. Hence the ghosts appear at the desired position for the beta approach aircraft. This visually simplifies the task of positioning the aircraft with the desired spacing. The paper provides further enhancements to the basic concepts of CDRA.

[0005] In Fig. 2 a concept of in-trail ghosting is illustrated. With in-trail ghosting, an in-trail ghost 20 is created from a real aircraft 14 on the alpha approach 10 to aid in the placement of another aircraft 14^1 on that approach.

[0006] At times, there may be gaps in the aircraft approaching on the alpha (primary) approach 10, such that it becomes necessary to resynchronize with the beta runway. A further enhancement illustrated in Fig. 3 provides for ghosting of aircraft 16 from the beta approach 12 back to the alpha approach 10 in order to aid in the "resynching" of air traffic. The ghost 24 is placed using a selectable spacing 26 back from the mirror position of aircraft 16.

[0007] The paper also introduces the concept of smart ghosting as illustrated in Fig. 4. With smart ghosting, an aircraft's weight category is taken into account, by providing a longer tie mode 28 and in trail 34 spacing for heavy aircraft 30 to generating tie mode 32 and in trail 36 ghosts allow for wake turbulence created by such aircraft.

[0008] The paper also mentions that a CDRA system has been operational at Calgary Alberta airport since May 2000.

[0009] Despite the disclosure of CRDA concepts and implementation at a single airport, many issues remain with regard to development of an air traffic control display aid adaptable to any geometry and configuration.

Summary of the Invention

[0010] An object of the present invention is to provide an improved visual aircraft spacing tool.

In accordance with an aspect of the present invention there is provided a method of visual aircraft spacing comprising the steps of: determining heading for an alpha approach; determining a heading for a beta approach; defining a target reference point in dependence upon the alpha and beta approaches; and determining an image reference point in dependence upon at least one of the target reference point, a difference between the headings, a characteristic of the beta approach and an offset.

[0011] In accordance with another aspect of the present invention there is provided an visual aircraft spacing tool comprising: a parameter for defining a heading for an alpha approach; a parameter for defining a heading for a beta approach; a module for defining a target reference point in dependence upon the alpha and beta approaches; and a module for determining an image reference point in dependence upon at least one of the target reference point, a difference between the headings, a characteristic of the beta approach and an offset.

Brief Description of the Drawings

[0012] The present invention will be further understood from the following detailed description with reference to the drawings in which:

Fig. 1 graphically illustrates a first mode of a known converging runway **display** aid (CRDA);

Fig. 2 graphically illustrates a second mode of the known CRDA of Fig. 1;

Fig. 3 graphically illustrates a third mode of the known CRDA of Fig. 1;

Fig. 4 graphically illustrates a fourth mode of the known CRDA of Fig. 1;

Fig. 5 illustrates in a functional block diagram an visual aircraft spacing tool (VAST) in accordance with an embodiment of the present invention;

Fig. 6 illustrates in a state diagram for a beta approach ghosting qualification region in accordance with an embodiment of the present invention;

Figs. 7a, b, c and d graphically illustrate runway configuration with reference points applied in accordance with an embodiment of the present invention;

Fig. 8 graphically illustrates a plurality of reference points in accordance with an embodiment of the present invention for an intersecting runway example;

Fig. 9a, b and c graphically illustrates tie mode ghosting for the example of Fig. 8;

Fig. 10a and 10b graphically illustrate in trail ghosting for the example of Fig. 8;

Figs. 11a, 11b, and 11c graphically illustrate resynchronization for the example of Fig. 8;

Figs. 12a and 12b graphically illustrate smart in trail ghosting by a known method and in accordance with an embodiment of the present invention;

Fig. 13 graphically illustrates track sequence lists in accordance with an embodiment of the present invention;

Figs. 14 and 15 graphically illustrates a high level of detailed data flow of track update process;

Fig. 16 graphically illustrates two overlapping qualification regions in accordance with an embodiment of the present invention;

Figs. 17a and 17b graphically illustrate direct route and indirect route examples of en-route VAST configurations in accordance with an embodiment of the present invention;

Figs. 18a and 18b illustrate data flow for activation and deactivation, respectively, of en-route VAST in accordance with an embodiment of the present invention;

Fig. 19 illustrates data flow for a track update in en-route VAST;

Fig. 20 graphically illustrates an example of a mirror ghosting projection in accordance with an embodiment of the present invention; and

Fig. 21 graphically illustrates an example of mirror ghosting on a single track in accordance with an embodiment of the present invention.

Detailed Description of the Preferred Embodiment

[0013] Referring to Fig. 5, there is illustrated in a functional block diagram main VAST data flow in accordance with

an embodiment of the present invention.

[0014] A main point of entry for the VAST processing is initiated with a track update 100. A track update processing *PlotUpdate()* routine performs three actions with respect to VAST:

1. Handles the drawing of any non-configuration based ghosting in trail targets.
2. Handles the recording of any trail dots required for VAST rendered targets.
3. Triggers the core VAST processing of each track for which an update is received. This is done implicitly through the *CheckTargetInGeographicFilters()* call 102.

[0015] The primary purpose of the check Target in Geographic Filter function with respect to VAST is simply to mark the track's attributes that indicate what VAST configuration, if any, the track is subject to processing by (Trk->CRDARPC) on the current update. Although the software allows for any number of active VAST configurations, an individual track can only be processed by a single VAST configuration at any one time.

[0016] The following is a high level overview of the VAST functionality handled by the ***CheckTargetInGeographic-Filters()*** 102 routine:

```
For each active geographical filter

      If the track is within the physical boundaries of
the region

            If  the track is within a link
                  mark the track as if it is within the
            main region[1]
            End if the track is in a linked region

            Switch(type of filter)

            Case Alpha Ghosting Qualification Region:
                  If CRDA is not active[2]
                        Stop processing this filter
                  End if CRDA is not active
                  If the track is not already associated
to an alpha GQR[3]
                        Set the Trk->CRDARPC attribute to
this configuration name[4]
                  End if the track is not already
associated to an alpha GQR
                  If  a log entry is required[5]  and
                       the track is correlated and
                       matches all qualifiers

                        Make the log entry
            End log entry required

            Case Beta Ghosting Qualification Region:
                  If CRDA is not active
                        Stop processing this filter
                  End if CRDA is not active
                  If the track is not already associated
to a beta GQR[6]
                        Set the Trk->CRDABetaGQR
attribute to this filter name[7]
                  End if the track is not already
associated to a beta GQR
                  If the track is correlated
                        If the track meets all of the
region qualifiers
                              Mark the region as active[8]
```

```
                                End if the track meets the
qualifiers
                            End if the track is correlated

                        End Switch on filter type

              End if the track is within the physical boundaries
of the region

End for each active geographical filter

TargetIsToBeGhosted()

DetermineCRDAAction()

CRDAProcessing()
```

[0017] As seen in the pseudo code illustrated above, the *CheckTargetInGeographicFilters()* terminates by calling *TargetIsToBeGhosted() 104, DetermineCRDAAction() 106* and *CRDAProcessing() 108* from the entry point into the main VAST logic.

[0018] The main purpose of the *TargetIsToBeGhosted()* function 104 is to determine if the current track requires any type of ghosting on this update. The track requires ghosting if one of the following conditions is met:

- The user has manually forced on a ghost for this aircraft (includes manual request of a Resynchronization ghost), or
- The track is correlated, within an alpha GQR, and matches the qualifications of that region.

[0019] As the following pseudo code illustrates, this is not quite as straightforward as it sounds:

```
If the track is within a total target filter
      Track is not to be ghosted - return FALSE
End if the track is within a total target filter

If the user has previously forced on a Resynchronization
ghost for this track
      If the track is not within a beta ghosting
qualification region
```

```
                Associate the track to the only active CRDA
configuration[1]
        End if the track is not within a beta GQR
        Track is to be ghosted - return TRUE [2]
End if a resync ghost exists for this track

If the track is not forced on
        If  the track is not within an alpha GQR or is not
correlated
                Track is not to be ghosted - return FALSE
        End track not within an alpha or not correlated
End track not  forced on

If the track is correlated and within an alpha GQR
        If the track matches the regions qualifiers
                If the track is forced ' primary'[3]
                        Mark the track as forced 'secondary'
                End if the track is forced 'primary'
                Track is to be ghosted - return TRUE
        End if the track matches the qualifiers
End if the track is correlated and within an alpha GQR


If the track has ghosting forced on
        If the track is not within an alpha GQR
                Associate the track with the only active
configuration
        End if the track is not within an alpha GQR
        Track is to be ghosted - return TRUE
End if the track has ghosting forced on

Track is not to be ghosted - return FALSE
```

[0020] The main purpose of the *DetermineCRDAAction()* function 106 is set the track's CRDAAction attribute to one of the following values:

- CRDA_NO_ACTION
- CRDA_CREATE_GHOST
- CRDA_MOVE_GHOST
- CRDA_DELETE_GHOST

```
If ghosting is required on this update

        Switch (current track CRDAAction setting)

        Case CRDA_NO_ACTION:
                New track CRDAAction = CRDA_CREATE_GHOST
```

```
            Case CRDA_DELETE_GHOST:
                  New track CRDAAction = CRDA_CREATE_GHOST

            Default:
                  New track CRDAAction = CRDA_MOVE_GHOST

            End switch on CRDA Action

      Else if ghosting is not required for this track

            Switch (current track CRDAAction setting)

            Case CRDA_CREATE_GHOST:
                  New track CRDAAction = CRDA_DELETE_GHOST

            Case CRDA_MOVE_GHOST:
                  New track CRDAAction = CRDA_DELETE_GHOST

            End switch on CRDA Action

      End if ghosting is not required
```

[0021] Once *TargetIsToBeGhosted() 104 and DetermineCRDAAction() 106* are used together to set the track's CRDAAction flag, CR*DAProcessing() 108* is called for this track This is the entry point to the heart of the VAST processing.

```
If forced ghosting or resync ghosting has been selected
for this track
      If the configuration that the track was forced for
is no longer active
            Override the track CRDAAction & set to
CRDA_DELETE_GHOST
      End if the configuration associated to this track
is no longer active
End if forced ghosting or resync ghosting has been
selected for this track


TrackSeqList_RemoveTrack()

Switch (CRDAAction)

Case CRDA_CREATE_GHOST:
      TrackSeqList_InsertTrack()
      CreateGhostTarget()
      DetermineInTrailGhostState()
      DrawGhostTargetBlock()

Case CRDA_CREATE_RESYNC_GHOST:
      CreateGhostTarget()
      DrawGhostTargetBlock()

Case CRDA_MOVE_GHOST:
      TrackSeqList_InsertTrack()
```

```
            EraseGhostTargetBlock()
            DetermineInTrailGhostState()
            DrawGhostTargetBlock()

      Case CRDA_DELETE_GHOST:
            EraseGhostTargetBlock()
            DeleteGhostTarget()

      End switch on CRDA Action
```

[0022] A function TrackSeqList_RemoveTrack() 110 tries to find the current track in a Track Sequence List 112 and remove it. On CRDA_MOVE_GHOST this track is inserted at a new position, as described below.

[0023] A function TrackSeqList_InsertTrack() 114 determines the distance of the aircraft from the Target Reference Point (TRP) for the active VAST configuration the track is within, as well as track wake turbulence classification for leading and trailing positions. Wake turbulence classifications are determined using a lookup table *AircraftTypeList* 116 or a flight plan if current aircraft type is not in the table. The *TrackSeqList_InsertTrack()* function 114 inserts this record into the Track Sequence List 112 at a position ordered by track distance from the TRP. Records in this list are updated every time the CRDA_MOVE_GHOST action is performed. The Track Sequence List 112 is used to determine the In-Trail Ghost Image Reference Point for in-trail ghosts.

[0024] All memory required by VAST is allocated dynamically. A CreatGhostTarget() function 118 allocates heap for the ghost target structure, initializes all ghost target attributes, and points the reference held within the parent target structure to this ghost record.

[0025] A DetermineInTrailGhostState() function 120 determines whether or not in trail ghosting is required for the specified parent track. Normally this is just determined by checking the VAST configuration settings related to the alpha Ghosting Qualification Region that the parent track is within. If the In Trail ghost is required and not currently allocated, then the appropriate memory is allocated and all structure elements are initialized. Conversely, if the In Trail ghost is allocated but is no longer required, this function will de-allocate the memory and initialize the In Trail ghost pointer within the ghost structure.

[0026] A DrawGhostTarget() function 122 controls the drawing of all ghost targets and their data blocks related to an individual track. This includes the tie / stagger / or resync ghost along with the in trail ghost. In addition, any PTL drawing required for a ghost target is also initiated from this function.

[0027] An EraseGhostTarget() function 124 removes all ghost targets from the plot window related to a single parent track. This includes removal of any PTLs that are being produced for these ghosts. As with the drawing mechanism for normal tracks, this function is often used in conjunction with *DrawGhostTarget()* 122 to move the ghost to a new position on the Plot Window.

[0028] A DeleteGhostTarget() function 126 removes any PTL associations, de-allocates all memory associated with the ghost targets, and resets all VAST attributes within the parent track structure.

[0029] Although the track update mechanism is the key trigger to initiate the VAST processing, there are several other entry points that will cause individual or all track's to be processed by this logic. The other events that can trigger the VAST processing include:

- Changing any value in an individual track's flight plan will cause that track to be re-processed against all active Ghosting Qualification Regions
- Activation or deactivation of VAST configurations will cause all tracks to be processed immediately against all active GQRs
- Activation or deactivation of VAST as a whole (from the QAB) will cause reprocessing of all tracks
- Selecting 'Forced Ghost / Resync Ghost' from the track menu will cause VAST processing to be performed for that specific track
- Manually removing a ghost via the ghost track's context menu

[0030] The *DrarvGhostTarrget()* function 122 supports two distinct drawing modes, a quick re-draw mode and a full draw mode. These are identified by CRDA_QUICK_DRAW and CRDA_FULL_DRAW respectively. Typically, the quick draw is used when it is desired to re-draw the ghost track and related data block in the same position as it was last drawn, the full draw mode is used otherwise.

[0031] First the symbol for the current ghost is chosen, then for a full draw the following functions are invoked:

- GetGhostPosition() 130,
- DetermineGhostTagContents() 132, and
- PositionTargetBlockElements() 134

[0032] *GetGhostPosition()* 130 calculates the new position for the ghost target in question, *DetermineGhostTagContents()* 132 sets up the ghost data block based on the current user selections. *PositionTargetBlockElements()* 134 is used to determine where to place the data block and how to attach it to the PPS with leader lines.

[0033] While *GetGhostPosition()* 130 and *DetermineGhostTagContents()* 132 are functions built specifically for VAST, *PositionTargetBlockElements()* 134 is the same function that is called for both normal tracks and ghosts. VAST makes use of routines that exist for normal tracks wherever possible. This is done by calling *GhostToTrackEntry()* to convert the ghost structure into a track structure, and then calling the routine that expected a track structure as one of its arguments. When the function returns, the *TrackEntryToGhost()* routine converts the modified track structure back to a ghost structure.

[0034] This philosophy was employed to prevent duplicating many functions that already existed for a track, when the same service was required by a ghost. In addition to reducing the need for duplicated code, this method prevents having to continually check within a function to see if we are dealing with a normal track or a ghost track, and prevents all functions from having to understand what comprises a ghost structure. This makes the code more maintainable, as any changes made to a single location affects both the ghost and the normal track.

[0035] After calling *PositionTargetBlockElements() 134,* if an In Trail ghost is required it is drawn before completing the rendering of the ghost target itself by calling *DrawInTrailGhostTargetBlock() 136.*

[0036] If we are in the quick draw mode, *DrawPTL()* (not shown in Fig. 5) is called to re-display the PTL for the current ghost. If we are in the full draw mode then *LayoutAndDrawPTL()* 138 is called instead to both recalculated and then draw the related PTL.

[0037] Finally the symbol colour is set based on the current type of ghost and whether the tower or terminal colour set is selected, and the ghost is displayed.

[0038] The EraseGhostTarget() function 124 is specifically written to handle ghost targets. When a target update is processed by *PlotUpdate()* the target must be erased before it is redrawn at the new position. A target block is also erased if it has been dropped, or become stale, as called from *DeleteDroppedAndStaleTargets().*

[0039] To erase a ghost target block, the routine calls *PaintRectOnPlotWindow()* to rebuild the area of the Plot Window encompassed by the ghost target block's bounding rectangle.

[0040] Before erasing the ghost target block, this routine sets the 'Exclude' flag for the target's ghost Track List entry. This will prevent the target block itself from being redrawn when the rectangle for the target block is being built up. Similarly if the target has a PTL then they can also be flagged Excluded from being redrawn. After the erasure has been completed then these exclusions will be removed.

[0041] The only special handling of this function that differs from the basic design of the *EraseTargetBlock()* routine is that this function will also remove any In Trail ghost that exist for the same parent track. This is done because all ghost targets related to a single parent are always updated in pairs.

[0042] All ghost targets related to the same parent track are updated on the same cycle, therefore the *DrawGhostTarget ()* routine calls *DrawInTrailGhostTarget()* 136 whenever there is an In Trail ghost related to the same parent.

[0043] Ghost targets make use of the same *RecordTrailDot()* function that is used by normal tracks to create an entry in the trail dot table (refer to the 'Trail Dot Recording' section of this document).

[0044] Much like the normal tracks, a call to this *RecordTrailDot()* function is made during the *PlotUpdate()* cycle. The added complication with ghost targets is that, for a variety of reasons, the ghost target may actually not be displayed later on in the VAST processing. Therefore we must do some cursory checks to ensure that the trail dot is not created if the ghost will itself later be suppressed from the display.

[0045] The conditions that will cause VAST not to display a currently created ghost target are as follows:

- Ghost target is suppressed by the current VAST configuration settings,
- Ghost targets is manually forced off by the user,
- The Ghosting on Demand feature is turned on and there are no qualifying aircraft in the related Beta Ghosting Qualification Region

[0046] The number of trail dots displayed for a ghost target will always be equal to the current setting for normal tracks, for simplicity there is no separate option for number of ghost trail dots. It is however possible to independently turn on and off ghost trail dots.

[0047] Referring to Fig. 6, there is illustrated in a state diagram possible states for a beta geographic qualification region (Beta GQR) in accordance with an embodiment of the present invention. Special processing is required to provide the functionality required by VAST's Ghosting on Demand feature. This feature, when enabled, causes VAST to auto-

matically enable and disable itself based on the presence of aircraft approaching the beta runway.

[0048] This requires the system to understand when qualifying aircraft are within a VAST configuration's beta Ghosting Qualification Region.

[0049] To do this, a list of all active Beta GQRs is maintained along with the current state of the region. As shown in Fig. 6, there are three valid Beta GQR states:

- CRDA_BETAGQR_ACTIVE 150
- CRDA_BETAGQR_INACTIVE 152
- CRDA_BETAGQR_PENDING 154

[0050] Initially, all Beta GQRs are entered into the *BetaGQRList* as 'pending' 154. Every 5 seconds a function *MarkInactiveBetaGQRs()* takes all regions still in the 'pending' state and moves them to the 'inactive' state 152. A *MarkPendingBetaGQRs()* function is then called to move any currently 'active' regions to the 'pending' state 154.

[0051] When a track update comes in, and that track is detected to be within a beta GQR, that region is moved from its current state to the 'active' state 150.

[0052] When VAST needs to project a ghost image for a track on the Alpha Approach, and Ghosting On Demand is enabled, it first checks the state of the related Beta GQR. Only if the region is 'active' or 'pending' is the ghost target displayed. The ghost image is suppressed if the region is 'inactive'.

[0053] The main algorithms for VAST centre around those related to calculating ghost positions. The following provides a high level overview of the *GetGhostPosition()* function 130 and then go into the details of the VAST algorithms for specific ghost projections.

[0054] Although the *GetGhostPosition()* function 130 calculates positions for all types of ghost targets, each individual ghost position calculation is treated separately for clarity.

[0055] The GetGhostPosition() function 130 first determines what VAST configuration to use for the ghost position calculations. This is the VAST configuration related to the alpha GQR that the parent track is within. Next, this function determines what type of ghost VAST it is calculating a position for (resynchronization ghost vs. tie mode ghost vs. stagger mode ghost vs. in trail ghost) and then calculates the position for this type of ghost (in Plot Window x,y coordinates). This is done taking into consideration mirror ghosting and wake turbulence (smart ghosting) options for the VAST configuration that is being used to produce the ghosts. In addition to calculating the position for ghost targets, this function also calculates the heading of the ghost in question (speed for a ghost target is never calculated, as it is always equal to the speed of the parent aircraft).

[0056] Referring to Fig. 7a, b, c, and d, there are geographically illustrated four examples of runaway configuration with reference points applied in accordance with an embodiment of the present invention.

[0057] Before getting into the details of the position calculations for each type of ghost, it is first important to understand how VAST may be applied to an operational environment. For clarity these discussions will be limited to runways, but the same concept applies to other uses of VAST.

[0058] Referring to Fig. 7a, the most important parameters of a VAST configuration that are used in ghost position determination include:

1. The magnetic heading of the alpha approach 160
2. The magnetic heading of the beta approach 162
3. The Target Reference Point (TRP) 164
4. The Image Reference Point (IRP) 166

[0059] The magnetic heading of the approaches is straight forward, it is the difference between these angles that determines what rotation needs to be applied when calculating the ghost positions.

[0060] The starting point for all ghost positions is the position of the parent track. The first step is usually (this depends on the ghost type) to rotate the track position around the Target Reference Point (or TRP) 164 by the difference between the angles of the approaches. The next step is to translate that position by the difference between the TRP 164 and the IRP 166. In other words the parent track is judged against its distance from the TRP 164 and the corresponding rotated ghost is drawn relative to the IRP 166.

[0061] If VAST were only applied to intersecting runways, as shown in Fig. 7a, then only a single reference point would be required (the common point, or runway intersection point). Using the TRP / IRP combination does not restrict VAST to intersecting runways. It can be adapted to converging non-intersecting runways (Figs. 7b and 7c), and to parallel runways (Fig. 7d).

[0062] In addition it is possible to adapt VAST in a variety of ways to the same approach configuration. For example, if it is desired to ensure that when one aircraft is at the threshold of its runway, the aircraft on the other approach is back 2nm from its threshold, then adapting the TRP and IRP to be the thresholds of the runways simplifies the spacing values

that must be specified. On the other hand if it is desired to ensure that each successive aircraft to reach the intersection is separated by a certain distance, then adapting the TRP and IRP to be the runway intersection would be more appropriate.

[0063] Referring to Fig. 8, there is graphically illustrated an example of reference points for intersecting runway configuration. Ghost images are typically offset from the IRP or TRP by a specified spacing. This spacing value changes dependent on ghost type, mirror ghosting, and wake turbulence considerations. The IRP depicted above (the one specified as part of the VAST configuration) is the main Image Reference Point. To reduce the processing required to compute the ghost positions, a series of image reference points are pre-calculated (using the TRP and IRP specifications and the current spacing values) when a VAST configuration is activated or when spacing values are manually changed by the user.

[0064] There are four pre-calculated Image Reference Points 180 used for Tie mode ghosting, and five pre-calculated Image Reference Points used for In Trail Ghosting 180.

[0065] The Image Reference Points for Tie Mode Ghosting 170 are Default 172, light 174, medium 176, and heavy 178. The Image Reference Points used for in Trail Ghosting 180 are default 182, Heavy-heavy 184, heavy medium 186, medium light 188, and heavy-light 190. The positions of these points as shown are determined based on the various spacing values comprising an individual CRDA configuration. These relative positions are by no means fixed.

[0066] The use of these reference points is further discussed in the following descriptions of ghost position calculations on a type by type basis.

[0067] Referring to Fig. 9a, b, and c, there are graphically illustrated generating a tie mode ghost in accordance with an embodiment of the present invention.

[0068] The most common ghost calculation is that of the tie mode ghost. This type of ghost is used as a 'bulls-eye' reference point for aircraft on the beta approach 162. When an aircraft on the alpha approach 160 reaches the Target Reference Point 164, the ghost it is producing is offset a specified spacing from the Image Reference Point 166.

[0069] For example, as shown in Fig. 9a, assume an airport has intersecting runways, and the TRP 164 and the IRP 166 are defined to be the intersection of those runways. If the tie mode spacing is set to two nautical miles (assume for simplicity that smart ghosting is disabled) then when an aircraft on the alpha runway reaches the intersection of the runways, the ghost for this aircraft is on the beta approach back two nautical miles. If this ghost is used by the controller as a bulls-eye for the beta approach aircraft then successive aircraft to reach the intersection are separated by 2nm.

[0070] To calculate the position for the tie mode ghost, we start with the position 200 of the parent track that is to produce the ghost. This position is rotated around the Target Reference Point 164 n degrees, where n is the difference between the magnetic heading of the two approaches. To do this we translate the whole picture such that the Target Reference is our new origin and use the standard rotation matrix on the translated track position as shown in Fig. 9b. To arrive at the final tie mode ghost position 202 we translate the rotated position by the appropriate Image Reference Point 166 as shown in Fig. 9c.

[0071] The chosen Image Reference Point 160 that is applicable for our current ghost calculation is the Default Tie Image Reference Point 172 unless smart ghosting is enabled. If smart ghosting is enabled then the IRP is chosen based on the Wake Turbulence (WT) category of the aircraft producing the ghost.

[0072] For stagger mode ghost calculations, the same method as described above is used with the main IRP 166 being used for the final translation rather than an IRP (for stagger mode there is no additional offset, i.e. the distance from the ghost to the main IRP is equal to the distance between the real aircraft and the TRP).

[0073] The heading calculation for both the stagger ghost and the tie ghost is the same. There are two formulas used for this calculation, one is used when Mirror Ghosting is enabled, and a different one is used when Mirror Ghosting is not enabled.

```
If Mirror Ghosting  is enabled

      Ghost Heading = Beta Approach Angle -
            (Parent Track Heading - Alpha Approach Angle)

else

      Ghost Heading = Beta Approach Angle +
            (Parent Track Heading - Alpha Approach Angle)
```

[0074] The In Trail ghost is the simplest of all VAST ghosts. Because the In Trail ghost is not projected from one approach onto another, there is no rotation applied to an In Trail ghost. The Mirror Ghosting and the concept of 'tie' and

'stagger' mode ghosting are not applicable to the In Trail ghosts.

[0075] As shown in Fig. 8, in order to calculate the In Trail ghost position it is necessary to choose an Image Reference Point. Unless Smart Ghosting is enabled, this point will always be the 'Default In Trail IRP' 182.

[0076] If Smart Ghosting is enabled the Image Reference Points 180 are calculated based on values adapted in the VAST configuration for four combinations of wake turbulence classes, considering leading and trailing aircraft:

- Heavy aircraft is followed by heavy aircraft 184,
- heavy aircraft is followed by medium aircraft 186,
- medium aircraft is followed by light aircraft 188
- heavy aircraft is followed by light aircraft 190,

[0077] For other combinations of wake turbulence classes or when there is no trailing aircraft the Default In Trail spacing is applied. If aircraft wake turbulence class is undefined the maximum adapted spacing value is always applied (i.e. when heavy aircraft is followed by light aircraft), assuming the worst-case scenario. All described wake turbulence combinations and examples of adapted in-trail spacing values are summarized in the following two Tables.

Table A - Minimum In Trail Spacing Values Available in the CRDA Configuration Panel

Trailing Aircraft	Leading Aircraft			
	Heavy	**Medium**	**Light**	**Unknown**
Heavy	Heavy/Heavy Stagger	Default Stagger	Default Stagger	Heavy/Heavy Stagger
Medium	Heavy/Medium Stagger	Default Stagger	Default Stagger	Heavy/Medium Stagger
Light	Heavy/Light Stagger	Medium/Light Stagger	Default Stagger	Heavy/Light Stagger
Unknown	Heavy/Light Stagger	Medium/Light Stagger	Default Stagger	Heavy/Light Stragger

Table B - Sample In Trail Spacing Values (in nautical miles)

Trailing Aircraft	Leading Aircraft			
	Heavy	**Medium**	**Light**	**Unknown**
Heavy	3	4	4	3
Medium	5	4	4	5
Light	7	6	4	7
Unknown	7	6	4	7

[0078] Some aircraft types are considered "special" in a sense that the different wake turbulence classification must be applied depending on whether an aircraft is in leading or in trailing position. "Special" aircraft types with their wake turbulence classes are supplied in the "aircraft.typ" file.

[0079] The algorithm that determines an Image Reference Point first consults the Aircraft Types List loaded from the "aircraft.typ" file. If aircraft is not a "special" aircraft then the wake class defined in the flight plan is used. If aircraft wake class is undefined, the worst-case spacing is applied as described above.

[0080] Connecting the Target Reference Point and the chosen Image Reference Point forms a vector. It is this vector that is added to the parent aircraft's position in order to determine the position for the In Trail ghost.

[0081] The heading for the In Trail ghost does not need to be calculated, as it is always equal to that of the parent aircraft.

[0082] As shown in Figs. 10a and 10b the target reference point (TRP) 164 is translated to the origin and the aircraft position 210 are translated by the same vector 212 to a position 210'. Next an image reference point 180 is selected, has the origin referenced to the it by a negative vector 212' as shown in Fig. 10b, the same vector 212' is then used to translate the position 210' to form an in trail ghost position 214.

[0083] The Smart In Trail Ghost Position calculations make use of the Aircraft Types List, which defines the "special" aircraft types that have a different wake turbulence category when in leading and in trailing position. The Aircraft Types List can only be used if the Smart In Trail Ghosting feature is enabled.

[0084] Each record in the Aircraft Types List contains the following fields:

- Aircraft type name
- Wake turbulence class in leading position
- Wake turbulence class in trailing position

[0085] The presence of the "aircraft.typ" file is not mandatory for Smart In Trail Ghosting operation. If "aircraft.typ" file is not present, the wake turbulence category specified in the track Flight Plan is used to define the in trail spacing. In this case the wake turbulence category of the aircraft in the leading and trailing position is the same.

[0086] Referring to Figs. 11a, b, and c, there is graphically illustrated the calculation of the Image Reference Point (IRP) for Resynchronisation ghost. The calculation uses a spacing value selected by the user from a cascade menu of resynchronisation spacing values, which is attached to the track pop-up menu. The projected resynchronisation ghost position is then obtained by rotating the parent track position 220' from beta onto alpha approach 222' as shown in Fig. 11b and translating the rotated position back 222 by difference between calculated IRP and main IRP, as shown in Fig. 11c.

[0087] After the Image Reference Point 224 is determined, the resync ghost position 222 is calculated in fundamentally the same way that the Tie Mode ghost is determined. The only difference is that the rotation is applied in the reverse direction. There are two formulas used for the calculation of the resync ghost's heading. One is used when Mirror Ghosting is enabled, and a different one is used when Mirror Ghosting is not enabled.

```
If Mirror Ghosting  is enabled

      Ghost Heading = Alpha Approach Angle -
            (Parent Track Heading - Beta Approach Angle)
else

      Ghost Heading = Alpha Approach Angle +
            (Parent Track Heading - Beta Approach Angle)
```

[0088] The known 'Smart In-Trail Ghosting' feature in the CRDA described with regard to Fig. 4 herein above uses the wake turbulence classification of the leading aircraft only in order to determine the spacing to be used when projecting the In-Trail ghost images for convenience this is shown in Fig. 12a. However, the mix of different aircraft classes and larger percentage of heavy aircraft in various airports, as well as greater demands on these airports creates a need to consider the wake class of both the leading and the trailing aircraft as shown in Fig. 12b. In addition, some aircraft types like B757 and P3 are defined as "special" types which means they are considered "heavy" aircraft in the leading position and "medium" aircraft in the trailing position. Thus for VAST a decision was made to define such aircraft types and their WT classes for leading and trailing position in a separate "*aircraft.typ*" binary file (Table C). This file is loaded into the AC Type table used for maintenance of the Track Sequence Lists.

Table C - Format of the "aircraft.typ" File

Aircraft type identifier	WT Class when leading	WT Class when trailing
P3	H	M
B757	H	M
...	...	...
...	...	...
WT Class notation: H - "heavy", M - "medium", L - "light".		

[0089] The Smart In-Trail Ghosting feature that considers the wake turbulence classification of both the leading and trailing aircraft, is implemented using a track sequence list created for each active CRDA configuration. Entries in the track sequence list are ordered according to the track distance from the Target Reference Point defined for that CRDA configuration (ascending order) and contain the following fields:

Table D - Track Sequence List Entry

Field Name	Data Type	Description
TrackID	unsigned int	Unique track identifier

(continued)

Field Name	Data Type	Description
Distance	Double	Track distance from TRP (nmi)
WTClassLeading	Char	Wake turbulence classification for aircraft when in leading position
WTClassTrailing	Char	Wake turbulence classification for aircraft when in trailing position

[0090] The following are valid values for the internal wake turbulence classifications of the aircraft (WTClassLeading and WTClassTrailing parameters in the record defined above):

"+" heavy
"/" medium
"-" light
" " (space) unknown

[0091] The definition of the VAST configurations has been modified in order to support different combinations of wake classifications of the leading and trailing aircraft as shown herein above in Tables A and B.

[0092] A Track Sequence List is maintained as a part of each active VAST configuration

[0093] Before a new track is added to the Track Sequence List, the WT classes (leading and trailing) are checked against the AC Type table based on aircraft type in the Flight Plan. If current aircraft type can not be found in the AC Type table, the WT Class field of the Flight Plan is used. If this field is not defined, the Unknown WT class is used (i.e., "***heavy***" when leading, "***light***" when trailing).

[0094] The maximum number of tracks to be sequenced is set to 40 per approach (a predefined constant). If the track sequence list is full and the distance of a new qualifying track does not exceed the maximum distance in the track sequence list, then the new track is inserted according to its distance and the farthest track is removed from the track sequence list. If the track sequence list is full and a new qualifying track is the farthest track compared to those in the track sequence list, then the new track is not inserted into the track sequence list.

[0095] The default in-trail stagger is applied to the last (i.e. unpaired) track in the sequence list.

[0096] The default in-trail stagger is applied to all qualifying tracks not inserted into the track sequence list.

[0097] If a new track happens to be at the same distance from Target Reference Point as an existing one, then new track is inserted into the track sequence list after the track with the same distance from Target Reference Point.

[0098] Non-correlated tracks are not sequenced.

[0099] Image reference points are pre-calculated using adapted values in order to speed-up processing, as described above with reference to Fig. 8.

[0100] The user is able to change adapted in-trail spacing values. Image Reference Points are re-calculated accordingly.

[0101] The high-level and detailed data flows of the track update process related to this enhancement are shown in Fig. 14 and Fig. 15 respectively.

[0102] The DetermineITSmartGhostIRP() routine in selects the proper Image Reference Point based on the WT classes of the current aircraft and the following aircraft sequenced on this approach. The code fragment for the Image Reference Point selection is shown below. For VAST weight constants used in the code refer to Wake Turbulence Classification.

```
static void
DetermineITSmartGhostIRP(
 Track *track,     /* parent track related to CRDA track
*/
 float *Xrb,     /* returned X position of IRP
*/
 float *Yrb)      /* returned Y position of IRP
*/
{
 CRDA_CONFIGURATION_IRP_STRUCT *pIRPRecord;
 CRDATrackS *GhostTarget;
 char NextWTClass;     /* Weight class of the trailing
aircraft */
 char CurrentWTClass;      /* Weight class of the current
aircarft */
```

```
 /*
  * Create a pointer to the correct ghost target structure (the
index of
  * entry within the array of ghost targets is the same as the
index
  * used for the track into the array of track structures.
  */
GhostTarget = &CRDATracksDb[track->trackId];
/*
 * Initialization.
 */
pIRPRecord = &CRDA_ConfigurationIRPRecord[GhostTarget->alphaGQR];
/*
 * Check if there is a Flight Plan for this track.
 */
if (!track->flags.isCorrelated)
{
     /*
      * Flight plan does not exist for this track - WT class is
unknown.
      * Log the condition and let the switch statement below deal
with the
      * logic.
      */
     ErrHdl_LogVa( ERRHDL_LOGWARNING, "DetermineITSmartGhostIRP",
                    "Could not determine WT class for leading track.
                      "Track ID: %i", track->trackId);
}
/*
 * Determine WT classification code for leading and trailing
aircraft
  * using track sequence list
  */
CurrentWTClass = TrackOrder_GetWTClassOfCurrentAC(track);
NextWTClass = TrackOrder_GetWTClassOfTrailingAC(track);
/*
  * Determine Image Reference Point based on certain combinations
of
 * WT classes of the leading and trailing aircraft
 */
switch (CurrentWTClass)
{
case CRDA_LIGHT_WEIGHT:
     switch(NextWTClass)
     {
      case CRDA_LIGHT_WEIGHT:
      case CRDA_MEDIUM_WEIGHT:
      case CRDA_HEAVY_WEIGHT:
      case CRDA_UNKNOWN_WEIGHT:
      case CRDA_NO_TRAILING_TRACK:
           *Xrb = pIRPRecord->InTrailDefaultIRP.x;
           *Yrb = pIRPRecord->InTrailDefaultIRP.y;
           break;
      default:
           *Xrb = pIRPRecord->InTrailDefaultIRP.x;
           *Yrb = pIRPRecord->InTrailDefaultIRP.y;
          break;
     }
     break;
 case CRDA_MEDIUM_WEIGHT:
      switch(NextWTClass)
      {
       case CRDA_LIGHT_WEIGHT:
```

```
                *Xrb = pIRPRecord->InTrailMediumLightIRP.x;
                *Yrb = pIRPRecord->InTrailMediumLightIRP.y;
                break;
          case CRDA_UNKNOWN_WEIGHT:
                *Xrb = pIRPRecord->InTrailMediumLightIRP.x;
                *Yrb = pIRPRecord->InTrailMediumLightIRP.y;
                break;
          case CRDA_MEDIUM_WEIGHT:
          case CRDA_HEAVY_WEIGHT:
          case CRDA_NO_TRAILING_TRACK:
                *Xrb = pIRPRecord->InTrailDefaultIRP.x;
                *Yrb = pIRPRecord->InTrailDefaultIRP.y;
                break;
default:
               *Xrb = pIRPRecord->InTrailDefaultIRP.x;
               *Yrb = pIRPRecord->InTrailDefaultIRP.y;
               break;
        }
        break;
 case CRDA_HEAVY_WEIGHT:
        switch(NextWTClass)
        {
         case CRDA_LIGHT_WEIGHT:
                *Xrb = pIRPRecord->InTrailHeavyLightIRP.x;
                *Yrb = pIRPRecord->InTrailHeavyLightIRP.y;
                break;
case CRDA_MEDIUM_WEIGHT:
                *Xrb = pIRPRecord->InTrailHeavyMediumIRP.x;
                *Yrb = pIRPRecord->InTrailHeavyMediumIRP.y;
                break;
case CRDA_HEAVY_WEIGHT:
                *Xrb = pIRPRecord->InTrailHeavyHeavyIRP.x;
                *Yrb = pIRPRecord->InTrailHeavyHeavyIRP.y;
                break;
case CRDA_UNKNOWN_WEIGHT:
                *Xrb = pIRPRecord->InTrailHeavyLightIRP.x;
                *Yrb = pIRPRecord->InTrailHeavyLightIRP.y;
                break;
case CRDA_NO_TRAILING_TRACK:
                *Xrb = pIRPRecord->InTrailDefaultIRP.x;
                *Yrb = pIRPRecord->InTrailDefaultIRP.y;
                break;
default:
               *Xrb = pIRPRecord->InTrailDefaultIRP.x;
               *Yrb = pIRPRecord->InTrailDefaultIRP.y;
               break;
        }
        break;
 case CRDA_UNKNOWN_WEIGHT:
       switch(NextWTClass)
        {
         case CRDA_LIGHT_WEIGHT:
                *Xrb = pIRPRecord->InTrailHeavyLightIRP.x;
                *Yrb = pIRPRecord->InTrailHeavyLightIRP.y;
              break;
case CRDA_MEDIUM_WEIGHT:
                *Xrb = pIRPRecord->InTrailHeavyMediumIRP.x;
                *Yrb = pIRPRecord->InTrailHeavyMediumIRP.y;
                break;
case CRDA_HEAVY_WEIGHT:
                *Xrb = pIRPRecord->InTrailHeavyHeavyIRP.x;
                *Yrb = pIRPRecord->InTrailHeavyHeavyIRP.y;
                break;
```

```
case CRDA_SPECIAL_WEIGHT:
            *Xrb = pIRPRecord->InTrailHeavyMediumIRP.x;
            *Yrb = pIRPRecord->InTrailHeavyMediumIRP.y;
            break;
case CRDA_UNKNOWN_WEIGHT:
            *Xrb = pIRPRecord->InTrailHeavyLightIRP.x;
            *Yrb = pIRPRecord->InTrailHeavyLightIRP.y;
            break;
case CRDA_NO_TRAILING_TRACK:
           *Xrb = pIRPRecord->InTrailDefaultIRP.x;
            *Yrb = pIRPRecord->InTrailDefaultIRP.y;
           break;
     default:
          *Xrb = pIRPRecord->InTrailDefaultIRP.x;
          *Yrb = pIRPRecord->InTrailDefaultIRP.y;
          break;
      }
     break;
default:
     *Xrb = pIRPRecord->InTrailDefaultIRP.x;
     *Yrb = pIRPRecord->InTrailDefaultIRP.y;
     break;
 }
```

[0103] Currently the CRDA configurations are independent entities and must be activated and deactivated individually. An embodiment of the present invention provides association of the VAST configurations that allow activating/deactivating of a set of VAST configurations grouped logically at the adaptation time.

[0104] In order to implement the associative relationship in VAST configurations, an additional data field "AssociatedWithName" has been introduced into the VAST configuration. This field is set to the name of the "parent" VAST configuration at adaptation time. Only one level of association is allowed: each child can have only one parent, parent configurations having at least one child configuration cannot be associated with any other configuration. In the parent VAST configuration the 'AssociatedWithName' field must be blank.

[0105] Rules for activation and de-activation of associated VAST configurations:

- When a parent VAST configuration is activated (deactivated) all associated configurations are activated (deactivated) simultaneously and added to (removed from) the list of active configurations and corresponding map overlays.
- Only parent configurations are shown in the drop-down menu of the available VAST configurations under "VAST" button; the user can select and edit individual configurations associated with the parent VAST using the VAST dialog panel.

[0106] Rules for re-activation of associated VAST configurations when new configuration file is loaded:

- Re-activation of associated VAST configurations is based on the name and type of the parent VAST. If any parent VAST in the new file has the same name and type (Terminal, En-Route) as previously active parent VAST, it will be activated together with all its child VAST configurations, even if some or all of those child configurations are new.

Note: parent VAST configuration can not have child configurations of other type because RAHSTA makes it impossible to associate configurations of different types.

[0107] If previously active parent configuration is now a child of the new parent, neither the new parent nor the child will be activated.

- Settings are preserved only for previously active parent and child configurations. All new child configurations of the previously active parent configuration are activated with the default settings, as adapted.

[0108] Referring to Fig. 16 there are graphically illustrated two overlapping ghost qualification regions (GQR) as a further embodiment. At any one time an aircraft may qualify for only a single alpha GQR. The alpha GQRs of different VAST configurations are allowed to overlap. However, the track management module allows only a single set of ghosts (a single Stagger or Tie or Resynchronization ghost, along with an In-Trail ghost) to be generated for a single aircraft at

any point in time. To do this effectively, the VAST logic preserves the "continuity" of ghosting when an aircraft enters and leaves an overlapping region.

[0109] Implementation of overlapping GQR does not require any changes in the VAST configuration structure and RAHSTA. If track is within the multiple GQR, then track qualifiers are checked and the ***previous*** region this track has qualified for is set as a region this track is currently associated with.

[0110] The code fragment from Crda.c file shown therein below illustrates how the application keeps track of the previous qualification region using global flags that indicate if the region where track is detected can be overridden.

```
/*-------------------------------------------------------------------
-----------
|   Purpose:
|       This functions is called when the a track has been detected
within
|       a registered CRDA Alpha Ghosting Qualification Region AND
matches
|       all qualifiers related to the filter.
-------------------------------------------------------------------
---------*/
static void
qualifyingTrackInAlphaGQR(
    int                     event,      /* pivot event */
    Pvt_Pointer             userData,   /* Unused */
    Pvt_Pointer             callData )  /* track and geographical
filter */
{
    CRDATrackS *GhostTarget;
```

```
    GEOFILTER_CALLBACK_STRUCT *data = (GEOFILTER_CALLBACK_STRUCT
*)callData;

    /*
     * Create a pointer to the correct ghost target structure (the
index of
     * the entry within the array of ghost targets is the same as the
index
     * used for the track into the array of track structues.
     */
    GhostTarget = &CRDATracksDb[data->track->trackId];

    /*
     * It is only possible to be within a single alpha GQR
     * at any one point in time.
     */
    if (allowAlphaOverride == RSIT_TRUE)
    {
        /*
         * To preserve continuity, when a track is qualifying for
more than
         * one alpha GQR, we will always use the previous region that
it was
         * qualifying for (if it is still valid).
         */
         if ((GhostTarget->previousAlphaGQR == CRDA_NONE) ||
             ((GhostTarget->previousAlphaGQR != CRDA_NONE) &&
              (GhostTarget->previousAlphaGQR == data->key)))
         {
             allowAlphaOverride = RSIT_FALSE;
         }

        /*
         * Save an index to the master CRDA configuration record
         * related to this alpha Ghosting Qualification Region.
         */
        GhostTarget->alphaGQR           = data->key;
        GhostTarget->alphaQualifiersMet = RSIT_TRUE;
    }
}


/*-------------------------------------------------------------------
-----------
|   Purpose:
|        This functions is called when the a track has been detected
within
|         a registered CRDA Beta Ghosting Qualification Region AND
matches
|       all qualifiers related to the filter.
-------------------------------------------------------------------
---------*/
static void
qualifyingTrackInBetaGQR(
    int                     event,       /* pivot event */
    Pvt_Pointer             userData,    /* Unused */
    Pvt_Pointer             callData )   /* track and geographical
filter */
```

```
{
    CRDATrackS *GhostTarget;
    GEOFILTER_CALLBACK_STRUCT *data = (GEOFILTER_CALLBACK_STRUCT
*)callData;

    /*
     * Create a pointer to the correct ghost target structure (the
index of
     * the entry within the array of ghost targets is the same as the
index
     * used for the track into the array of track structues.
     */
    GhostTarget = &CRDATracksDb[data->track->trackId];

    if (allowBetaOverride == RSIT_TRUE)
    {
        if ((GhostTarget->previousBetaGQR == CRDA_NONE) ||
            ((GhostTarget->previousBetaGQR != CRDA_NONE) &&
             (GhostTarget->previousBetaGQR == data->key)))
        {
            allowBetaOverride = RSIT_FALSE;
        }

        GhostTarget->betaGQR           = data->key;
        GhostTarget->betaQualifiersMet = RSIT_TRUE;
    }

    /*
     * Although the track will not be considered 'officially'
     * within more than one beta GQR, it is possible for the
     * track to 'mark active' any number of beta GQRs that it
     * is within.
     */
    if (data->track->flags.isCorrelated)
    {
        /*
         * Track is within a Beta GQR, is correlated, and
         * matches all defined CRDA qualifiers related to
         * the current beta GQR - mark the region as active.
         * This will cause the ghosting on demand feature
         * to realize that there are tracks on the beta
         * approach.
         */
        GEOFILTER_SetState(CRDA_BetaFilterType, data->key,
GEOFILTER_STATE_ACTIVE);
    }
}



/*-----------------------------------------------------------------
-----------
|   Purpose:
|       This is a message handler for GEOFILTER_RESET_TRACK message.
-----------------------------------------------------------------
---------*/
static void
resetAllowOverride(void)
```

```
{
    allowAlphaOverride = RSIT_TRUE;
    allowBetaOverride = RSIT_TRUE;
}
```

Note: parameters previousAlphaGQR, previousBetaGQR are initialized to CRDA_NONE when new track entry is created in the track database;

[0111] Although the known CRDA allows for multiple configurations to be active at one time, each of these configurations calculates how far the parent track is off the centerline of the Alpha approach and displays ghost at the same distance from the Beta approach centerline. This restriction makes it difficult to use CRDA as an en-route spacing tool.

[0112] In accordance with an embodiment of the present invention, an en-route VAST module provides a controller with a pictorial representation of the relative positions of aircraft on divergent airways. For en-route VAST configurations the association mechanism is used for grouping configurations by a particular metering fix. All child configurations inherit the Image Reference Point (IRP) 166 from their parent. This IRP 166 is then used to draw a virtual Beta approach line and collapse ghosts onto it. Although the Target Reference Point (TRP) 164 for configurations within the group may differ, it is desirable that they all have a common TRP because track distance (and, as a result, ghost position) is calculated relative to the corresponding TRP.

[0113] Depending on whether the Alpha approach line leads directly to the fix or not, two types of en-route VAST configurations can be created:

- Direct route - any track qualifying for the region produces a ghost at a distance from the IRP being equal to the track distance from the TRP 164 as shown in Fig. 17a.

- Indirect route - any track qualifying for the region produces a ghost at a distance from the IRP being equal to the track distance from the TRP plus a distance correction see Trk2 as shown in Fig. 17b. The distance correction is obtained by interpolation of normalized track distance from the Route End Point between the Minimum Track Offset and Maximum Track Offset specified for this configuration see equation (1). Adaptation of indirect en-route CRDA configurations requires the following additional parameters:

  - Route End Point latitude and longitude - geographic location where indirect route ends
  - Minimum Track Offset (nmi) defines track distance correction at the Route End Point
  - Maximum Track Offset (nmi) defines track distance correction at the farthermost point of Alpha GQR relative to the Route End Point

$$GhostDistance = D + (MinimumTrackOffset + MaximumTrackOffset * (d / GQRLength)), \text{ (equation 1)}$$

where: GhostDistance - ghost distance from IRP, D - track distance from TRP, d - track distance from Route End Point, GQRLength - maximum length of GQR.

By combining both direct and indirect en-route configurations a user can define a single metering fix for airways of different characteristics and force all ghosts onto one Beta approach.

En-Route VAST configurations contain the "Heading Variance" parameter defining the minimum difference between the track heading and the region alpha approach heading when a heading indicator must be displayed on the projected ghost.

The visibility of the virtual Beta approach line and programmatically generated scale on the Display depends on the "ShowBetaApproach" parameter (YES/NO) of the En-Route VAST configuration. This parameter can be set only at adaptation time.

Implementation details:

[0114]

- Implementation of en-route VAST uses a separate map overlay for virtual Beta approach.
- The length of Beta approach line is calculated based on the maximum region size and the maximum value of all adapted offsets for associated indirect routes.
- The mileage scale for the virtual Beta approach can be created at run time, depending on the setting of the "Show-BetaApproach" parameter (see above). Scale steps are based on the length of the Beta approach line. If the length is greater than 50 nmi the smaller step is 10 nmi and the larger step is 50 nmi, otherwise the smaller step is 1 nmi and the larger step is 5 nmi.
- The ghost position is translated along the Beta approach line according to the calculated track distance. Since there is no ghost rotation involved, the same magnetic variation is applied to all defined IRPs in order to draw all virtual Beta approaches according to their adapted angles despite the distance between IRPs.
- Non-correlated tracks are not ghosted.
- The 'Stagger Display Mode' is enforced at all times.
- Ghosts can be forced on non-qualifying tracks through the track menu (as with the terminal VAST configurations).
- Mirror Ghosting and Resynchronization Ghosting options are not available for En-Route VAST configurations.
- Track sequence lists are not created for En-Route VAST configurations.
- A heading indicator is displayed on the projected ghost if the difference between the track heading and the heading specified for the region is more than the Heading Variance. If the Heading Variance has been set to zero at adaptation time, heading indicator is always displayed.

[0115] Data flow for En-Route VAST activation and deactivation is shown in Fig. 18. Data flow for track update in En-Route VAST is shown in Fig. 19.

[0116] Referring to Fig. 20 there is graphically illustrated an example of a mirror ghosting projection in accordance with an embodiment of the present invention. For simplicity, stagger mode ghosts are shown in a mirror projection. The advantage of the mirror projection is that aircraft can be vectored into the beta approach on a side remote form the alpha approach while simple rotation would result in the ghosts being on a side of the beta approach near the alpha approach.

[0117] However, in some situations it may be desirable for the controller to be able to turn on/off the mirror ghosting feature on an individual ghost track basis (independently of the current mirror ghost setting for a particular configuration). An example of mirror ghosting on individual track is shown in Fig. 21.

[0118] In order to keep track of the mirror ghosting state for individual targets two Boolean flags are introduced into GhostTrackS structure:
"allowMirrorGhostingOverride" and "isMirrorGhosted". The first flag is set to TRUE when a track qualifies for the region and is used to make the mirror ghosting menu option available. If the track qualifies, the second flag is initially set to the current Mirror Ghosting setting of the related VAST configuration. This flag is updated every time the user toggles mirror ghosting on the current ghost track. The mirror ghost position is calculated only if "isMirrorGhosted" flag is set to TRUE. In Fig. 20, stagger mode ghosts are projected for tracks 1 and 3 and a mirror ghost is used for track 2.

[0119] Non-correlated tracks can not be mirror-ghosted.

[0120] In order to simplify creation and processing of resynchronization ghosts a range of stagger values for a particular configuration must be specified at adaptation time and used to create a cascade menu on the AT display. This cascade menu is attached to the "Resync Ghost" button of the track menu.

[0121] For dynamic menu creation on the AT displays two extra parameters are introduced into the VAST configuration: Resync Min Stagger (nmi) and Resync Max Stagger (nmi). These values are positive integers (0 - 99) defining the lower and upper bounds for stagger values displayed in the menu. A Step of 1 nmi is used as an increment.

[0122] If Resync Min Stagger equals to Resync Max Stagger, a menu is not created and, as a result, the "Resync Ghost" option is not available in the track menu.

[0123] User selection of the resync stagger is stored in the "trackResyncSV" field of the CRDATrackS structure.

[0124] If the lowest resync stagger is greater than 0, the "0 nmi" button is added to the menu.

[0125] For non-correlated tracks Resynchronisation Ghosting is not available.

[0126] Further advantageous aspects of the present invention are set out in the following numbered clauses.

1. A method of visual aircraft spacing comprising the steps of :

   selecting a configuration type from an intersecting runways configuration, a; converging runways configuration; a parallel runways configuration; and an en-route configuration;
   determining heading for an alpha approach of the selected configuration type;
   determining a heading for a beta approach of the selected configuration type;
   defining a target reference point in dependence upon the alpha and beta approaches;
   determining an image reference point in dependence upon at least one of the target reference point, a difference between the headings, a characteristic of the beta approach and an offset;

selecting a mode of ghosting from tie mode, in-trail mode, and resynchronizing mode; and
determining a spacing used for separating a ghost image from an aircraft image in dependence upon the mode.

2. A method as set out in clause 1, wherein the spacing is dependent upon weight classes of both leading and trailing aircraft.
3. A method as set out in clause 1, or clause 2, further comprising establishing a plurality of configurations.
4. A method as set out in any of clauses 1 to 3, further comprising associating at least two of the plurality configurations.
5. A method as set out in clause 4, wherein the step of associating at least two of the plurality of configurations includes designating a single configuration as a parent.
6. A method as set out in any of the preceding clauses, wherein the step of selecting a configuration type selects an en-route configuration.
7. A method as set out in any of the preceding clauses, further comprising establishing a plurality of configurations having a particular metering fix.
8. A method as set out in clause 7, wherein the alpha approach leads directly to the particular metering fix.
9. A method as set out in any of the preceding clauses, wherein the step of selecting a mode of selects the re-synchronising mode.
10. A method as set out in any of the preceding clauses, wherein the spacing is dependent upon user selection from a predetermined list.
11. A method as set out in any of the preceding clauses, further comprising the step of selecting an image projection type.
12. A method as set out in clause 11, wherein the step of selecting an image projection type includes selecting from a normal and a mirror.
13. A method as set out in clause 12, wherein the step of selecting an image projection type selects a mirror.
14. A method as set out in clause 13, wherein the step of selecting an image projection type mirror applies to an entire region.
15. A method as set out in clause 13, wherein the step of selecting an image projection type mirror applies to a track.
16. A visual aircraft spacing tool comprising:

a computer display system;
a parameter for defining a heading for an alpha approach;
a parameter for defining a heading for a beta approach;
a module for defining a target reference point in dependence upon the alpha and beta approaches;
a module for determining an image reference point in dependence upon at least one of the target reference point, a difference between the headings, a characteristic of the beta approach and an offset;
a module for selecting a mode of ghosting from tie mode, in-trail mode, and re-synchronising mode; and
a module for determining a spacing used for separating a ghost image from an aircraft image in dependence upon the mode.

17. A visual aircraft spacing tool as set out in clause 16, wherein the spacing is dependent upon weight classes of both leading and trailing aircraft.
18. A visual aircraft spacing tool as set out in clause 16 or clause 17, further comprising a module for establishing a plurality of configurations.
19. A visual aircraft spacing tool as set out in any of clauses 16 to 18, further comprising a module for associating at least two of the plurality of configurations.
20. A visual aircraft spacing tool as set out in clauses 19, wherein the module for associating at least two of the plurality of configurations includes means for designating a single configuration as a parent.
21. A visual aircraft spacing tool as set out in any of clauses 16 to 20, wherein the configuration type is an en-route configuration.
22. A visual aircraft spacing tool as set out in any of clauses 16 to 21, further comprising a module for establishing a plurality of configurations having a particular metering fix.
23. A visual aircraft spacing tool as set out in any of clauses 16 to 22, wherein the alpha approach leads directly to the particular metering fix.
24. A visual aircraft spacing tool as set out in any of clauses 16 to 23, wherein the spacing is dependent upon weight classes of both leading and trailing aircraft.
25. A visual aircraft spacing tool as set out in any of clauses 16 to 24, wherein the mode is the re-synchronising mode.
26. A method as set out in any of clauses 16 to 25, wherein the spacing is dependent upon user selection from a predetermined list.
27. A method as set out in any of clauses 16 to 26, further comprising a module for selecting an image projection type.

28. A method as set out in clause 27, wherein the image projection type includes a normal and a mirror.
29. A method as set out in clause 28, wherein the image projection type is a mirror.
30. A method as set out in clause 29, wherein the image projection type mirror applies to an entire region.
31. A method as set out in clause 29, wherein the image projection type mirror applies to a track.

Claims

1. A method of visual aircraft spacing for en-route configurations comprising
determining a heading for a en-route parent alpha approach having a particular metering fix;
determining a heading for an en-route child alpha approach having the particular metering fix;
determining a target reference point for the parent alpha approach in dependence upon the approaches;
determining an image reference point for the parent alpha approach in dependence upon at least one of the target reference point, a difference between the parent alpha heading and a virtual beta heading, a characteristic of the parent alpha approach and a predetermined offset;
determining a virtual beta approach line utilising the determined image reference point for the parent alpha approach;
calculating a track distance equal to the distance between the target reference point and an aircraft present on the parent or child approaches;
utilising the calculated track distance to project a ghost image of the aircraft onto the virtual beta approach line.

2. A method according to claim 1 wherein the child alpha approach line leads directly to the metering fix.

3. A method according to claim 1 wherein the child alpha approach line comprises an indirect en-route configuration that does not lead directly to the metering fix.

4. A method according to claim 3 further comprising:

calculating a distance correction for the indirect en-route configuration; and

wherein utilising the calculated track distance comprises producing the ghost image at a distance from the image reference point equal to the track distance from the target reference point plus the distance correction.

5. A method according to claim 4 wherein the position of the ghost image relative to the image reference point is calculated according to the expression:

$$GhostDistance = D+(MinimumTrackOffset+MaximumTrackOffset *(d/GQRLength))$$

wherein
'GhostDistance' is the distance of the ghost image from the image reference point of the virtual beta approach;
'D' is the calculated track distance from the target reference point;
'd' is a track distance from a route end point said route end point being defined as the point where the indirect en-route configuration ends;
'GQRLength' is the maximum length of an alpha ghosting qualification region to which the indirect en-route configuration belongs;
'MinimumTrackOffset' is the track distance correction at the route end point; and
'MaximumTrackOffset' defines track distance correction at the farthermost point of the alpha ghosting qualification region relative to the route end point.

6. A method according to any preceding claim further comprising displaying a heading indicator on the projected ghost image if the difference between the approach and a heading specified for the region to which the approach belongs is greater than a specified value.

7. A method according to any preceding claim wherein all ghosts are forced to stagger mode and mirror and resynchronization ghosting options are not available.

8. A method according to any preceding claim wherein the length of the virtual beta approach line is calculated based

on a maximum region size and the maximum value of all adapted offsets for associated indirect configurations.

9. A method according to claim 8 wherein the mileage scale for the virtual beta approach line is created at run-time and the scale steps are based on the calculated length of the virtual beta approach line.

10. A method according to any preceding claim wherein a plurality of en-route child alpha approaches are determined.

11. A method according to claim 10 wherein a corresponding track distance is calculated for aircraft on any or all of the plurality of child alpha approaches and the corresponding track distance is used to project a ghost image of said aircraft onto the virtual beta approach line.

12. A method according to any preceding claim wherein the virtual beta approach line is displayed in a separate map overlay.

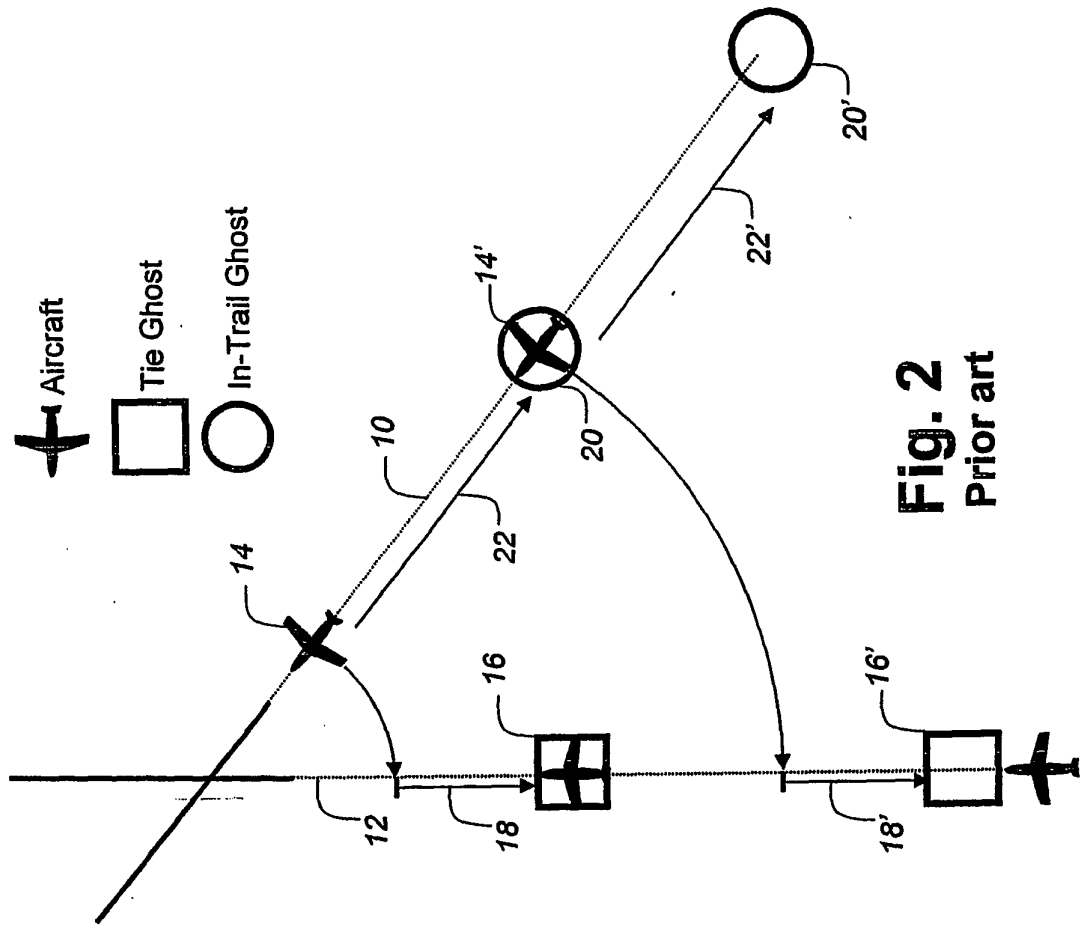


Fig. 2
Prior art

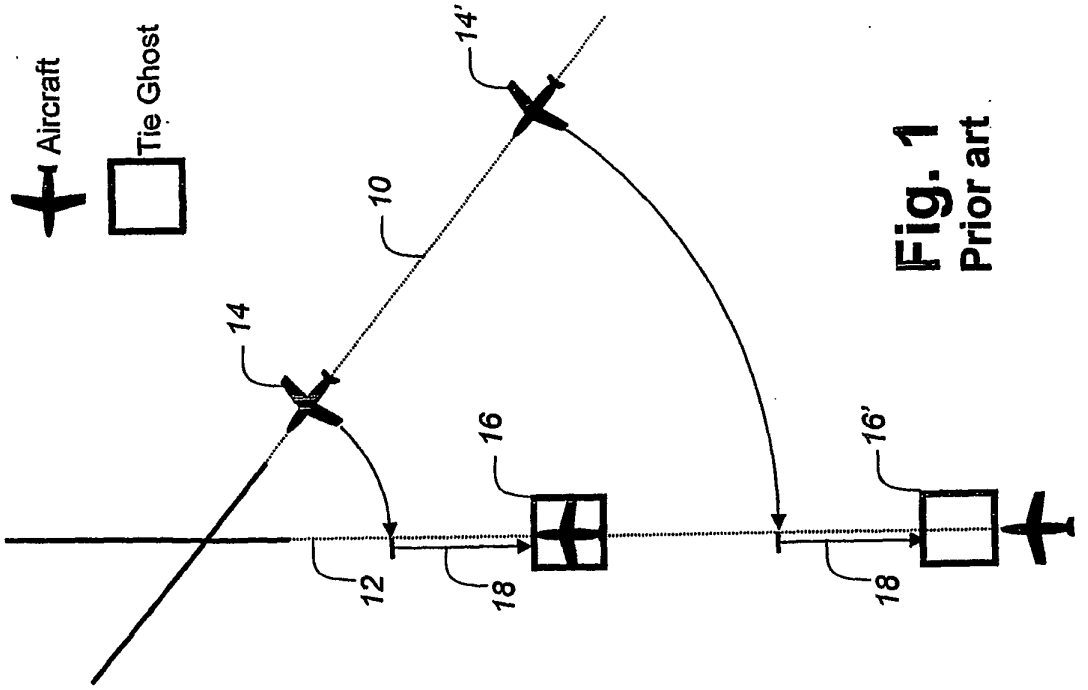


Fig. 1
Prior art

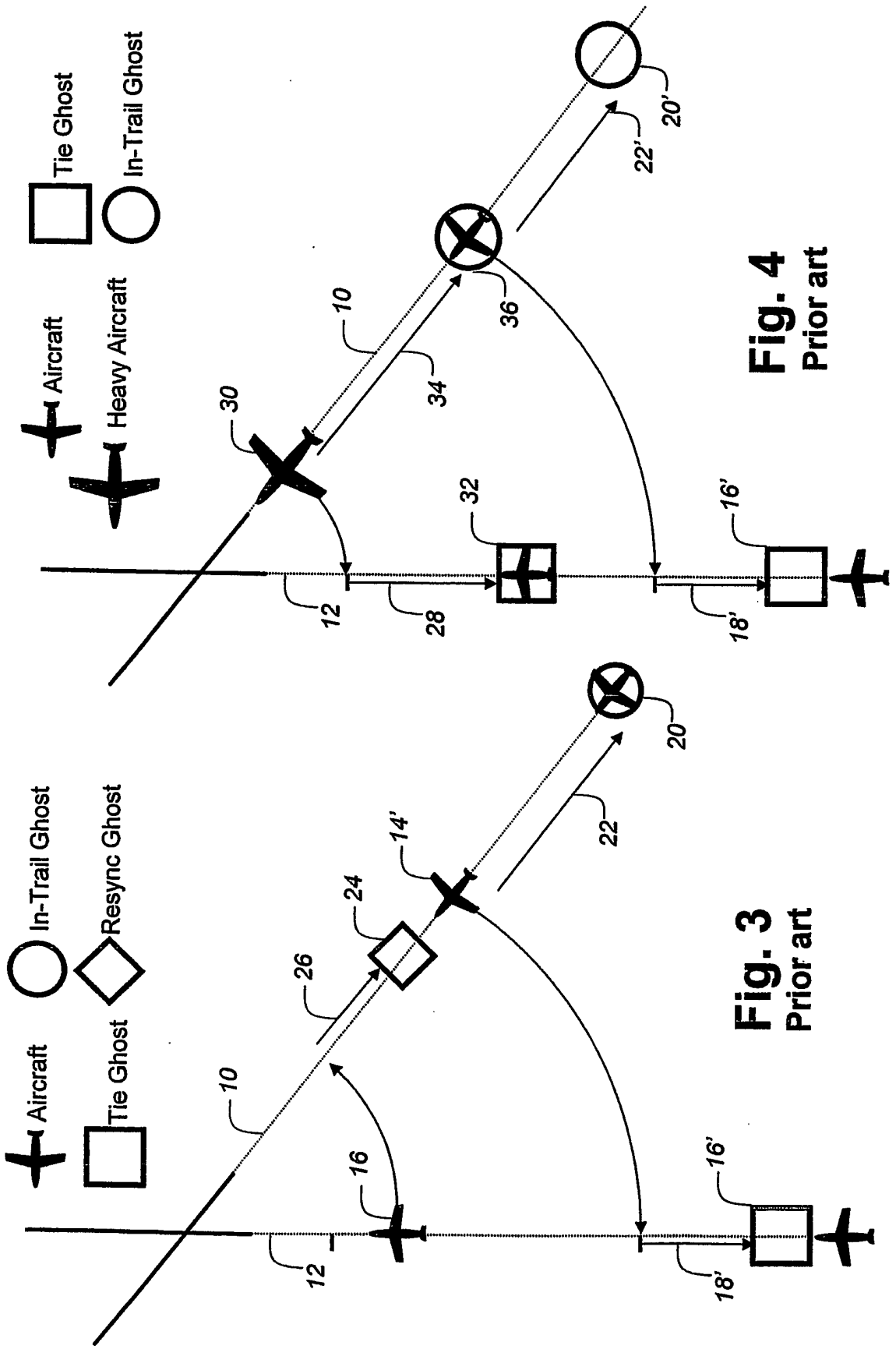
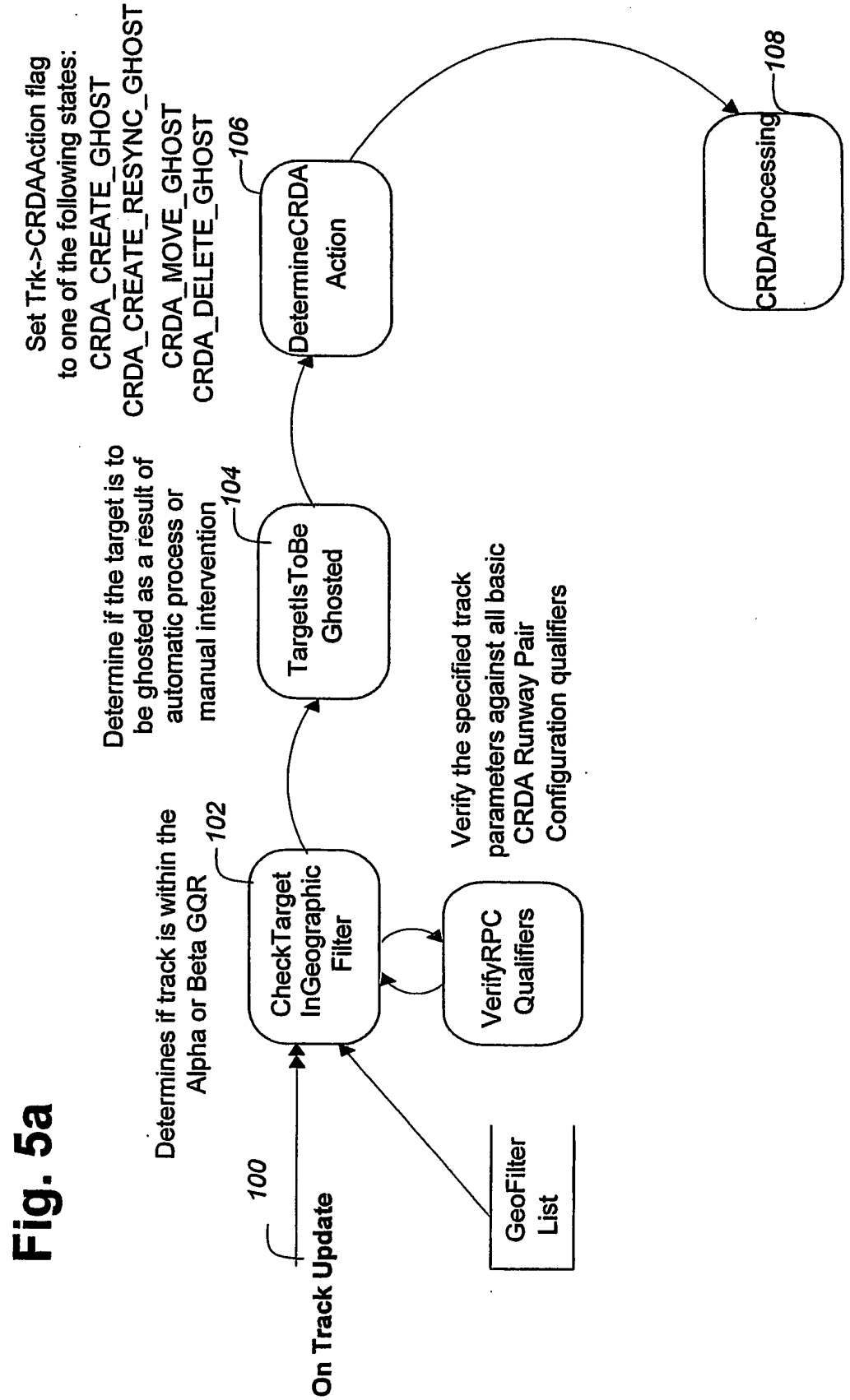


Fig. 4
Prior art

Fig. 3
Prior art

Fig. 5a


On Track Update
100
CheckTarget InGeographic Filter
102
Determines if track is within the Alpha or Beta GQR
TargetIsToBe Ghosted
104
Determine if the target is to be ghosted as a result of automatic process or manual intervention
DetermineCRDA Action
106
Set Trk->CRDAAction flag to one of the following states:
CRDA_CREATE_GHOST
CRDA_CREATE_RESYNC_GHOST
CRDA_MOVE_GHOST
CRDA_DELETE_GHOST
CRDAProcessing
108
VerifyRPC Qualifiers
Verify the specified track parameters against all basic CRDA Runway Pair Configuration qualifiers
GeoFilter List

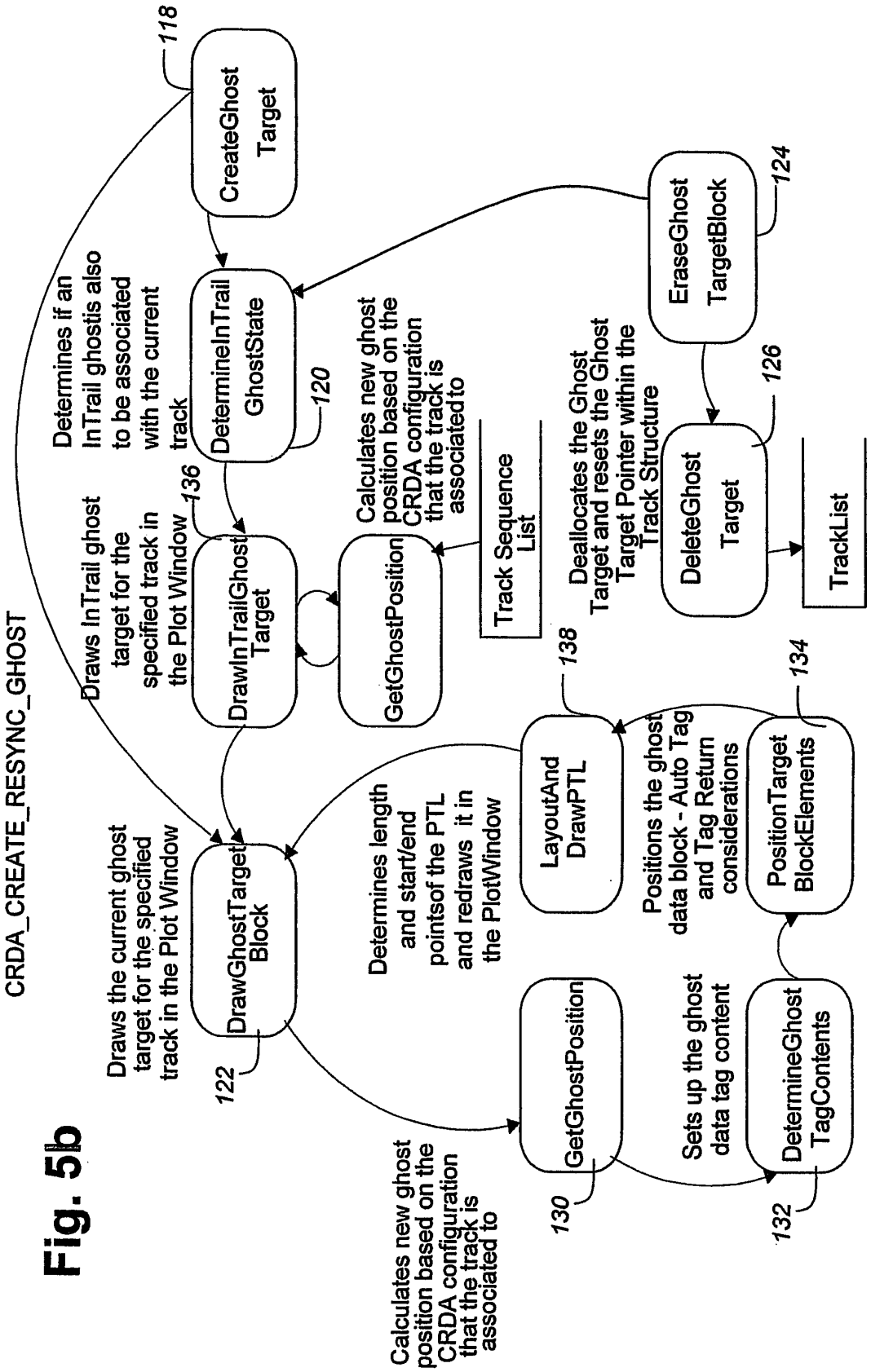

CRDA_CREATE_RESYNC_GHOST
CreateGhost Target
118
Determines if an InTrail ghostis also to be associated with the current track
DetermineInTrail GhostState
120
Draws InTrail ghost target for the specified track in the Plot Window
136
DrawInTrailGhost Target
GetGhostPosition
Calculates new ghost position based on the CRDA configuration that the track is associatedto
Track Sequence List
EraseGhost TargetBlock
124
Deallocates the Ghost Target and resets the Ghost Target Pointer within the Track Structure
DeleteGhost Target
126
TrackList
Draws the current ghost target for the specified track in the Plot Window
DrawGhostTarget Block
122
Determines length and start/end pointsof the PTL and redraws it in the PlotWindow
LayoutAnd DrawPTL
138
Positions the ghost data block - Auto Tag and Tag Return considerations
PositionTarget BlockElements
134
Calculates new ghost position based on the CRDA configuration that the track is associated to
GetGhostPosition
130
Sets up the ghost data tag content
DetermineGhost TagContents
132

Fig. 5b

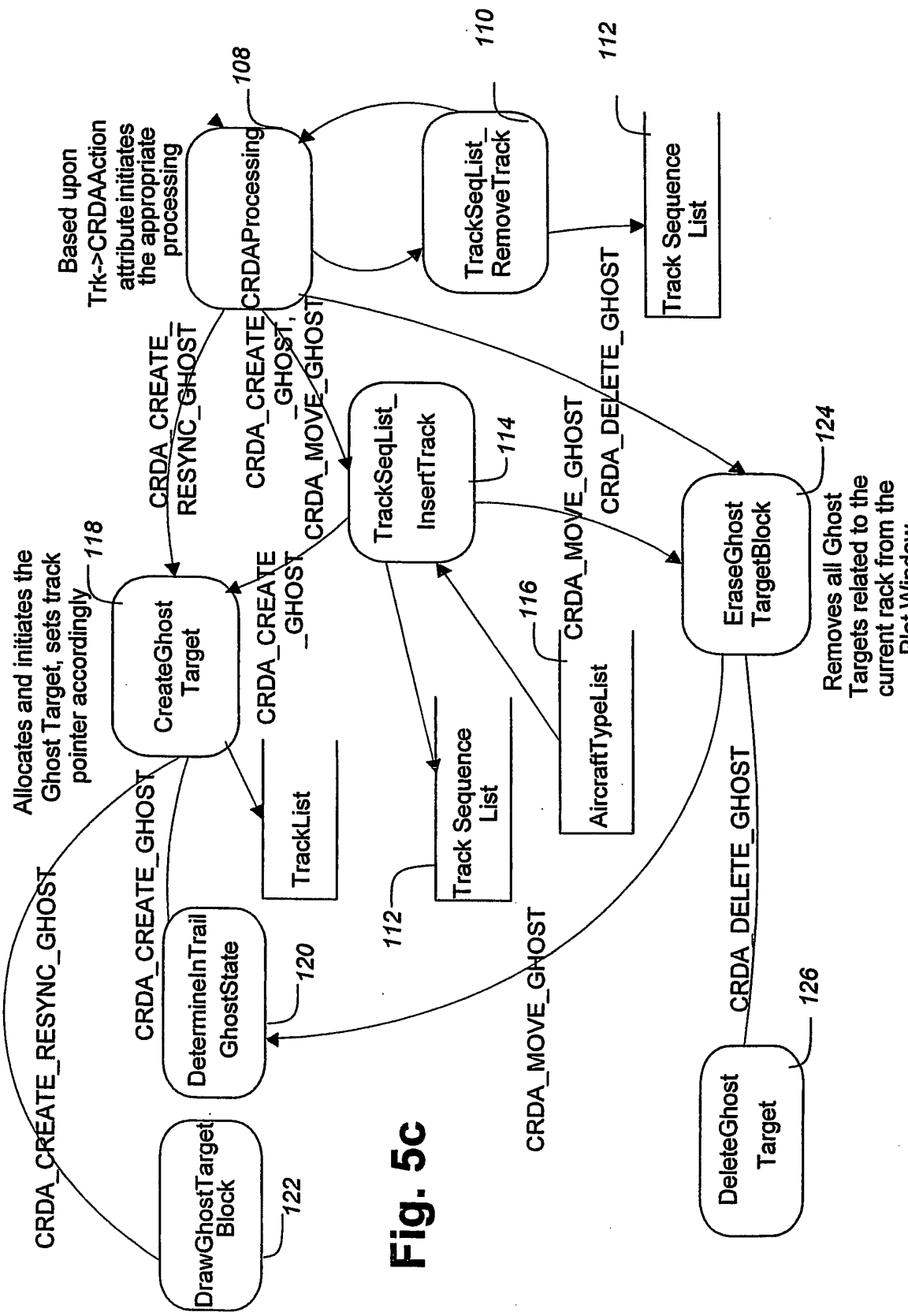

Based upon Trk->CRDAAction attribute initiates the appropriate processing
CRDAProcessing
108
TrackSeqList_ RemoveTrack
110
Track Sequence List
112
CRDA_CREATE_ RESYNC_GHOST
CRDA_CREATE_ GHOST, CRDA_MOVE_GHOST
CRDA_DELETE_GHOST
Allocates and initiates the Ghost Target, sets track pointer accordingly
118
CreateGhost Target
TrackSeqList_ InsertTrack
114
CRDA_CREATE_ GHOST
TrackList
CRDA_CREATE_GHOST
CRDA_MOVE_GHOST
116
AircraftTypeList
EraseGhost TargetBlock
124
Removes all Ghost Targets related to the current rack from the Plot Window
CRDA_CREATE_RESYNC_GHOST
DetermineInTrail GhostState
120
112
Track Sequence List
CRDA_MOVE_GHOST
CRDA_DELETE_GHOST
DrawGhostTarget Block
122
DeleteGhost Target
126

Fig. 5c

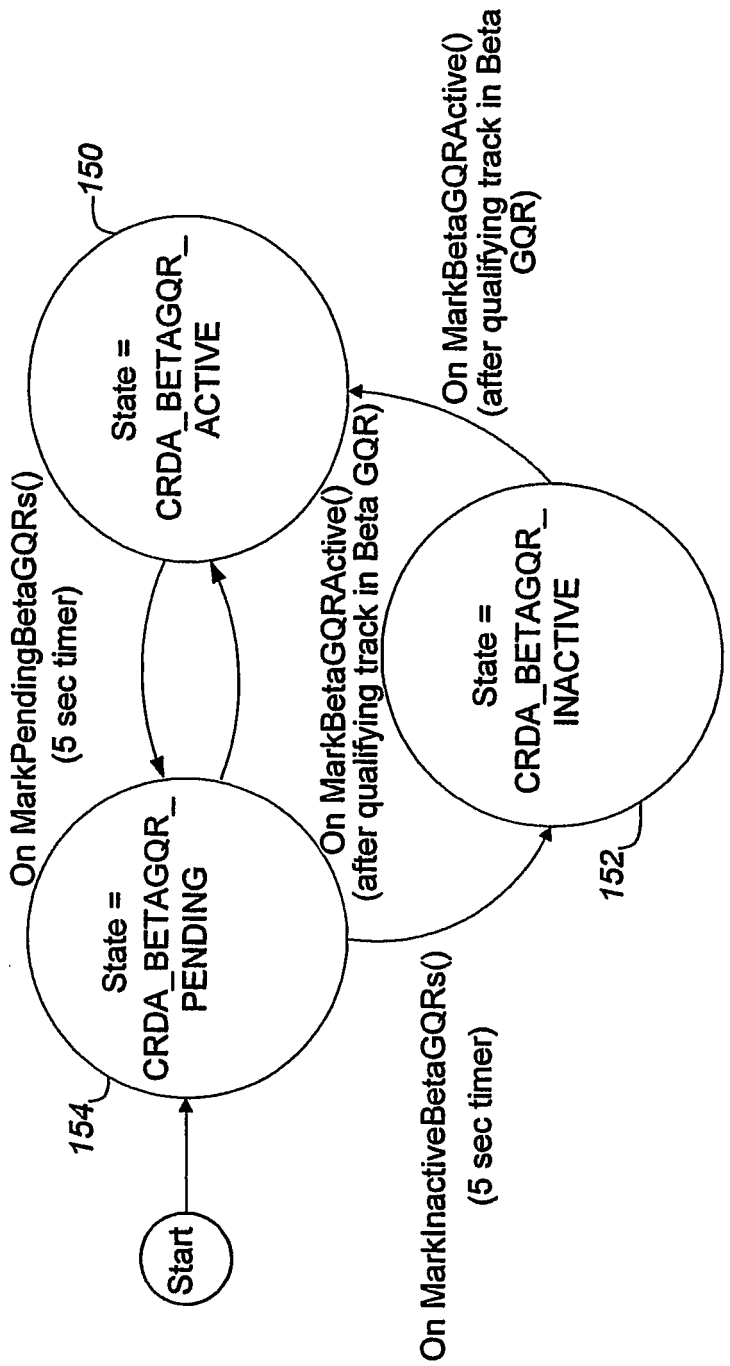

150
State = CRDA_BETAGQR_ ACTIVE
On MarkPendingBetaGQRs() (5 sec timer)
State = CRDA_BETAGQR_ PENDING
154
Start
On MarkBetaGQRActive() (after qualifying track in Beta GQR)
On MarkBetaGQRActive() (after qualifying track in Beta GQR)
State = CRDA_BETAGQR_ INACTIVE
152
On MarkInactiveBetaGQRs() (5 sec timer)

Fig. 6

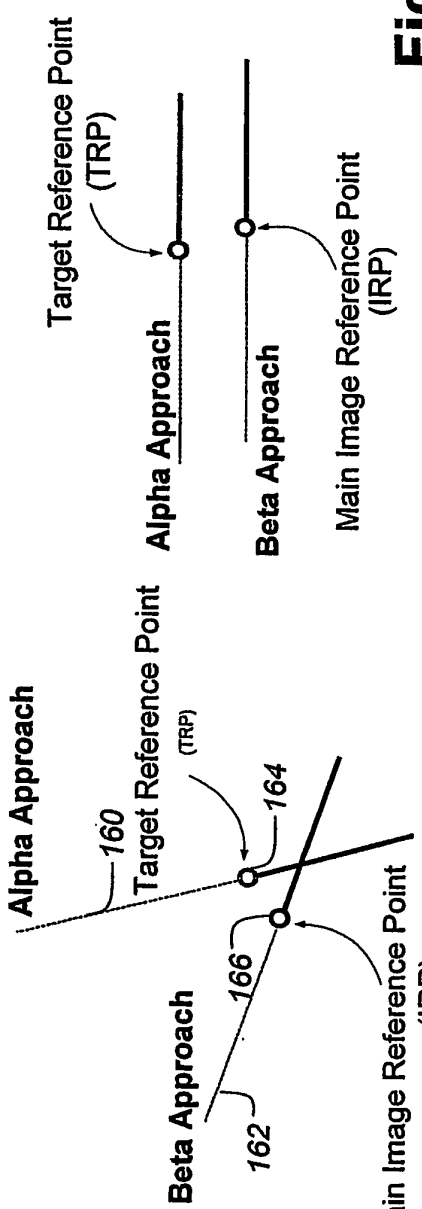


Fig. 7a

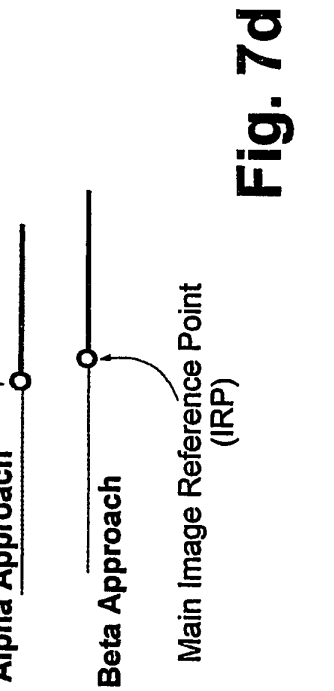


Fig. 7d

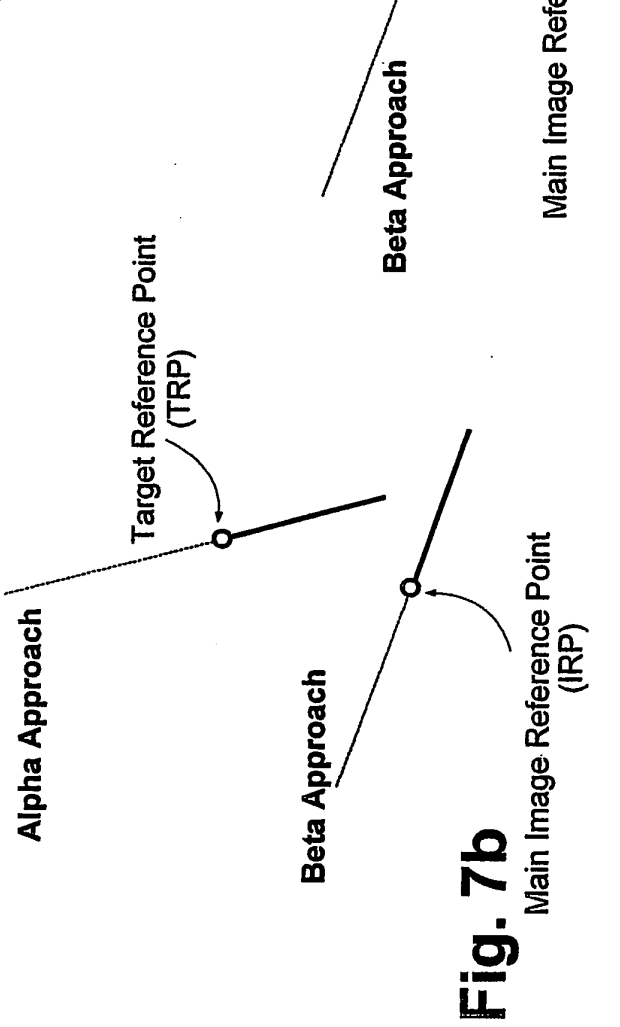


Fig. 7b

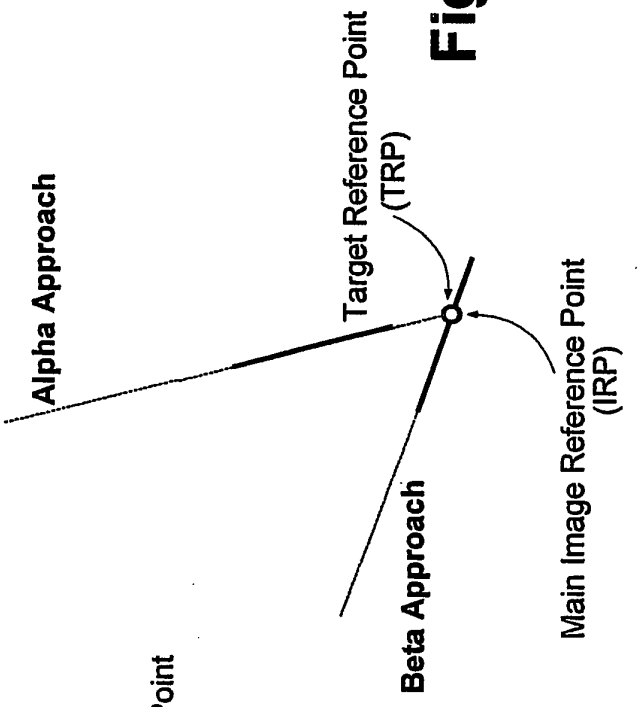


Fig. 7c

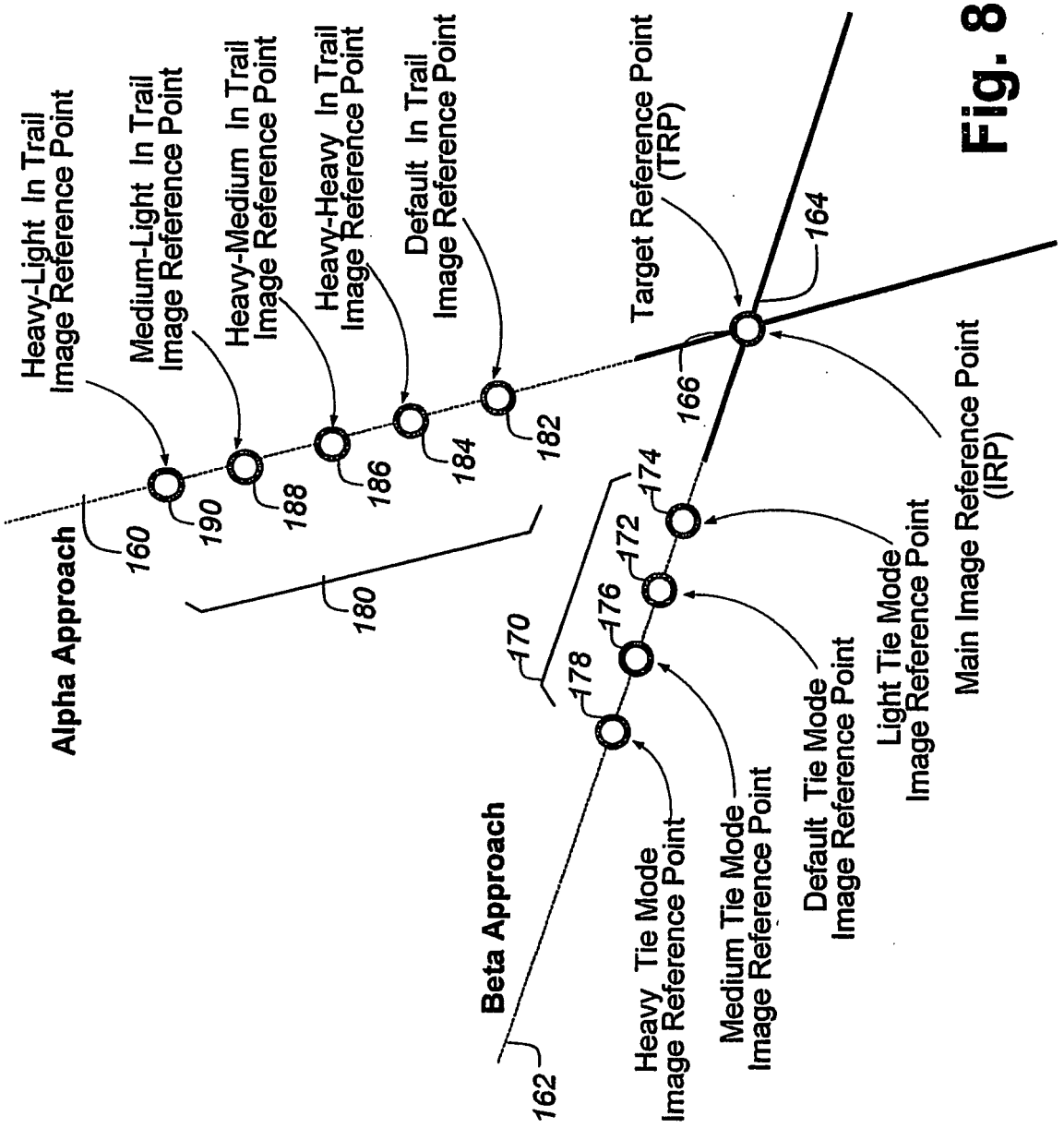
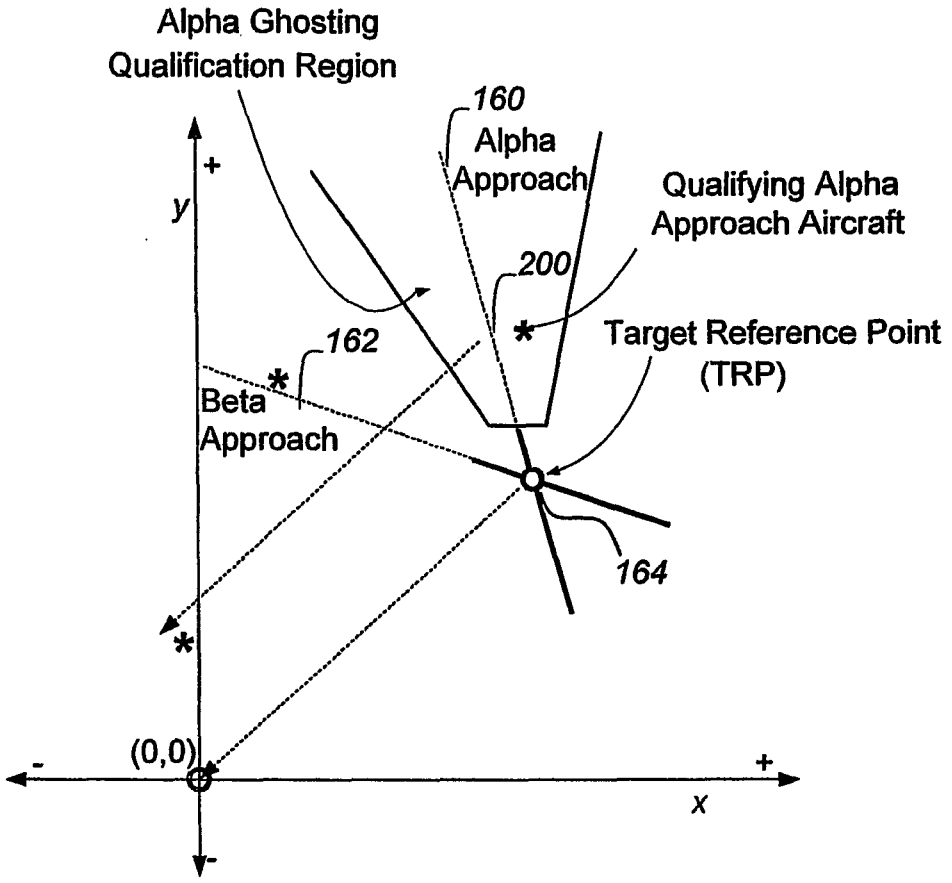

Heavy-Light In Trail Image Reference Point
Medium-Light In Trail Image Reference Point
Heavy-Medium In Trail Image Reference Point
Heavy-Heavy In Trail Image Reference Point
Default In Trail Image Reference Point
Target Reference Point (TRP)
Alpha Approach
Beta Approach
Heavy Tie Mode Image Reference Point
Medium Tie Mode Image Reference Point
Default Tie Mode Image Reference Point
Light Tie Mode Image Reference Point
Main Image Reference Point (IRP)
160
162
164
166
170
172
174
176
178
180
182
184
186
188
190

Fig. 8

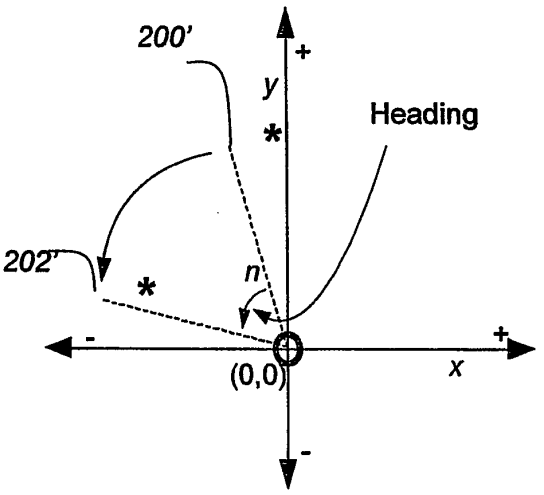


Step 1: TRP translated to the origin, the same translation is then applied to the original aircraft position.

Fig. 9a

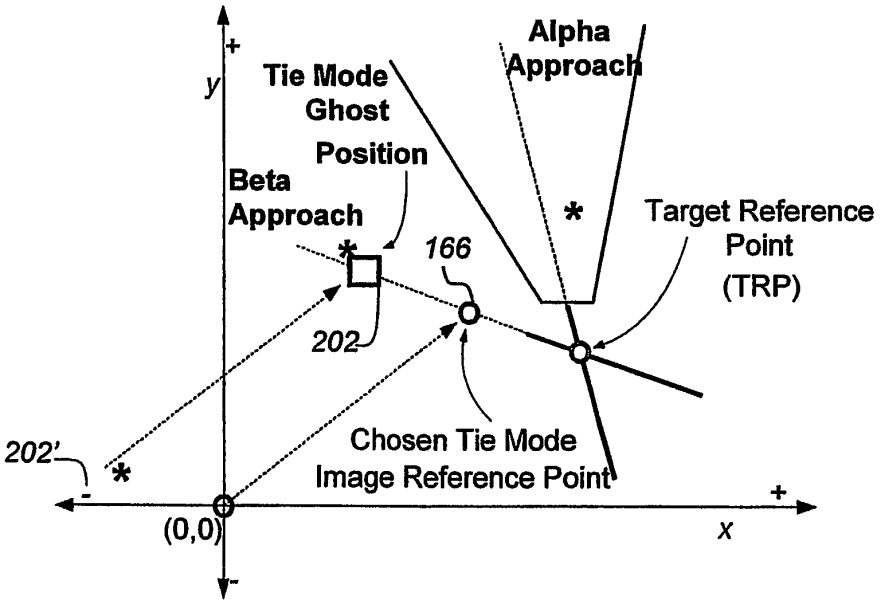
Fig. 9b

Rotation = Beta Approach
Heading - Alpha Approach



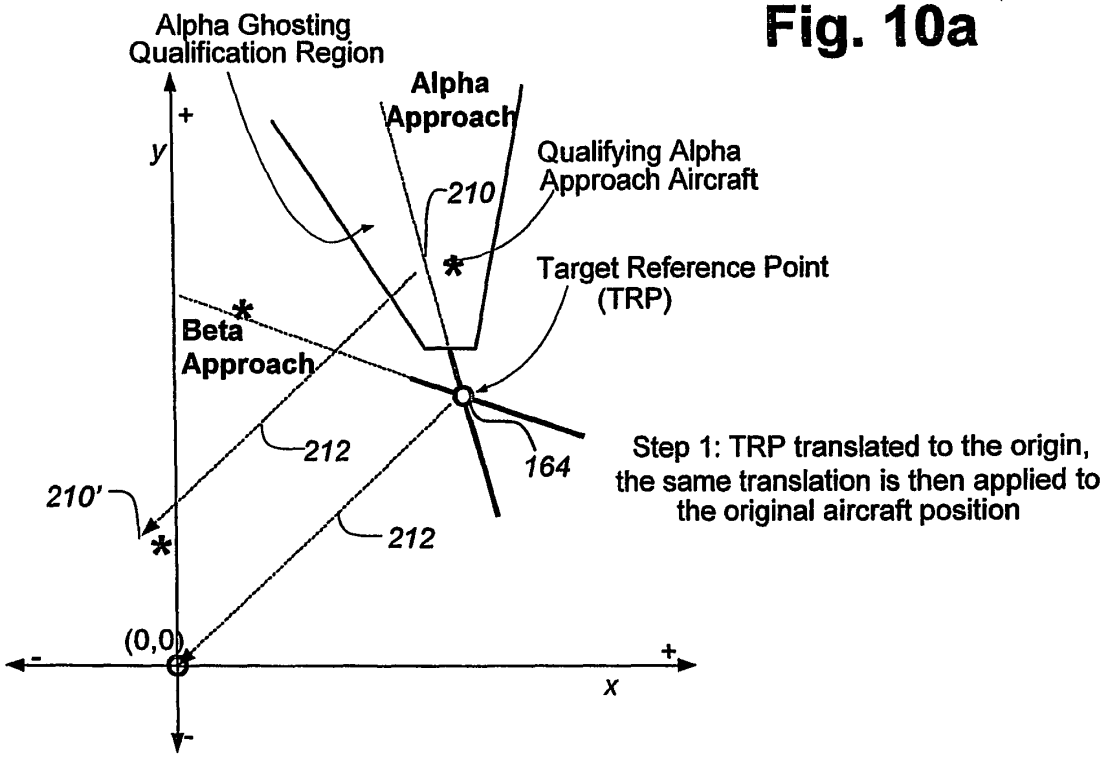
Step 2: Translated aircraft position is rotated around the origin by the difference between the two approaches.

Fig. 9c



Step 3: The chosen Image Reference Point is used to translate the rotated position to the final ghost position.

Fig. 10a



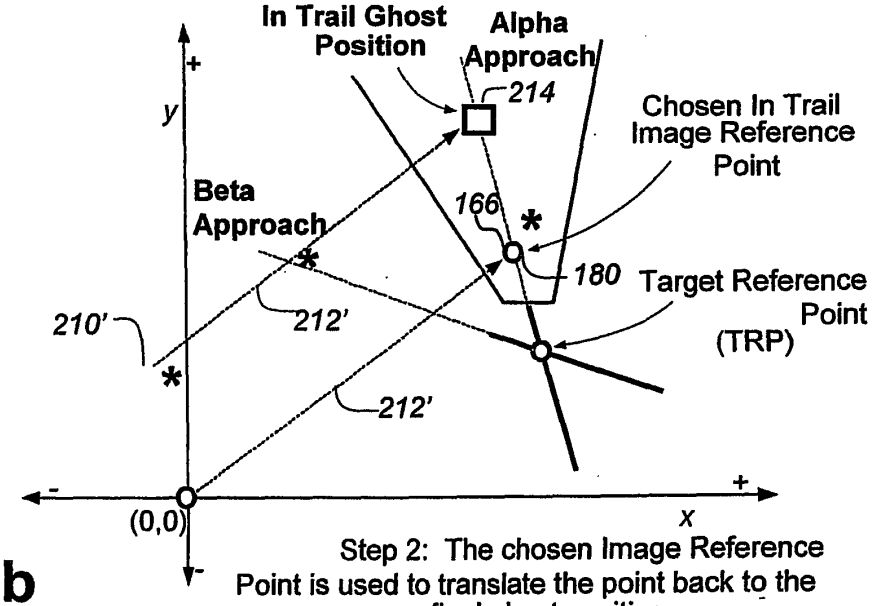
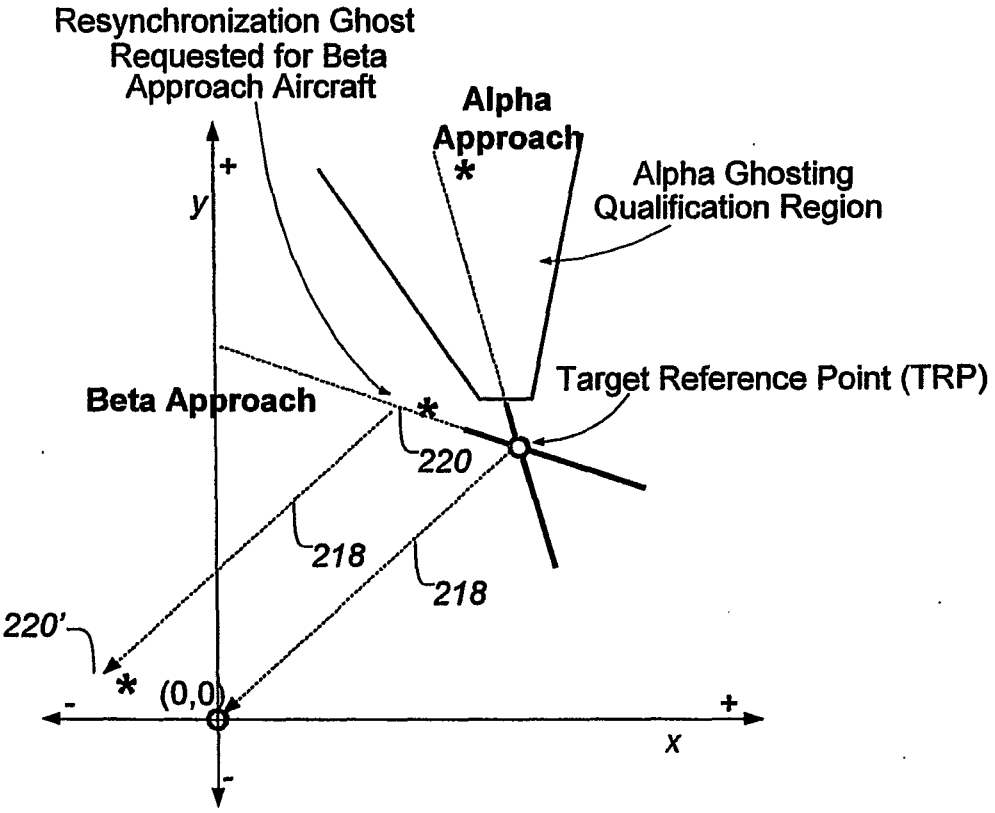


Fig. 10b



Step 1: Same translation required to move the Target Reference Point to the origin is used to translate the beta approach aircraft for whom the resynchronization ghost was requested.

Fig. 11a

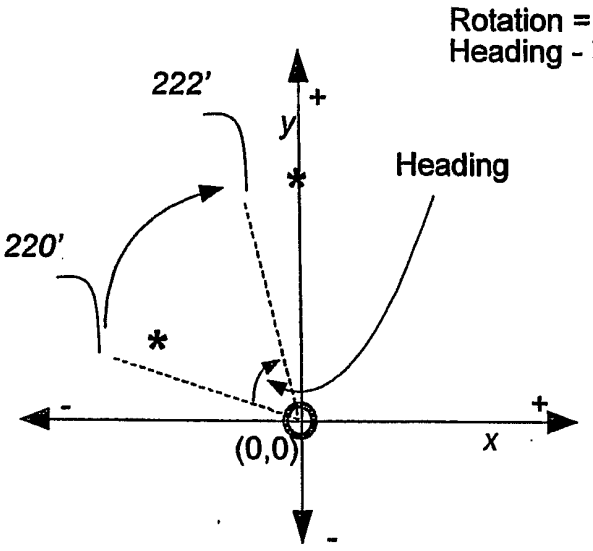
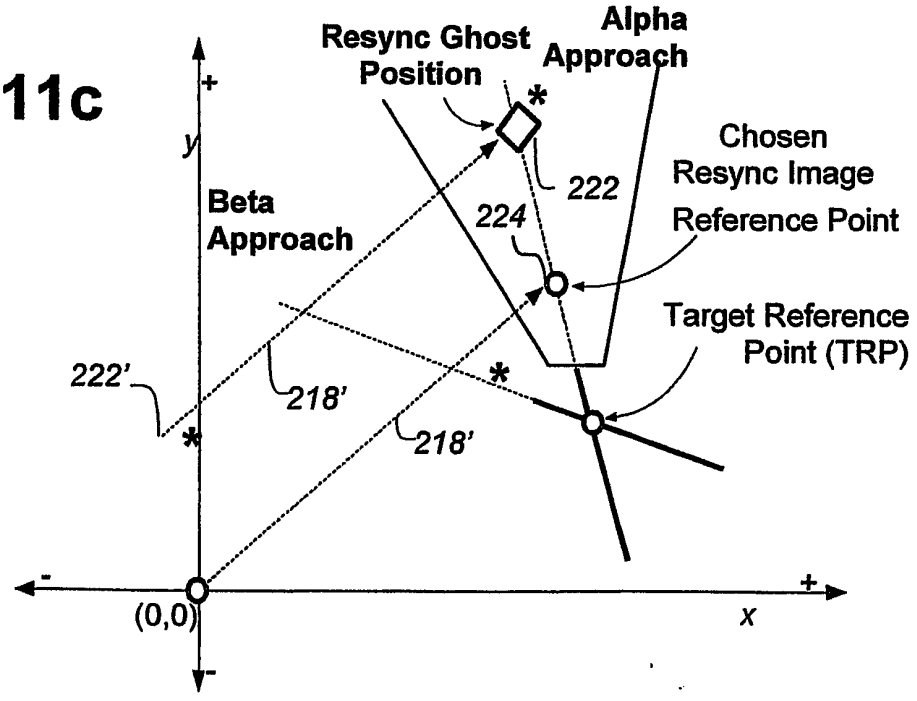


Fig. 11b

Step 2: Translated aircraft position is rotated around the origin by the difference between the two approaches in the opposite direction as the Tie Mode rotation.

Fig. 11c



Step 3: A Resync IRP is calculated using the spacing value selected by user from the Resynchronisation menu. The translation required to move the origin to this point is used to translate the rotated track position to a final ghost position.

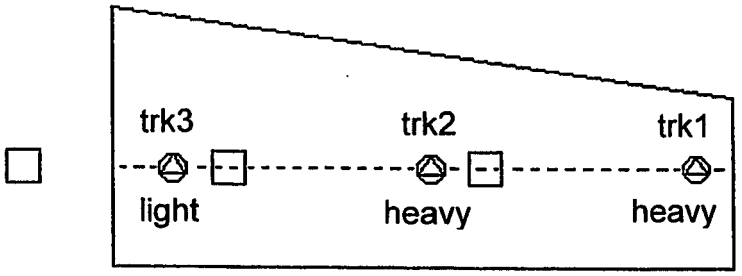


Fig. 12a

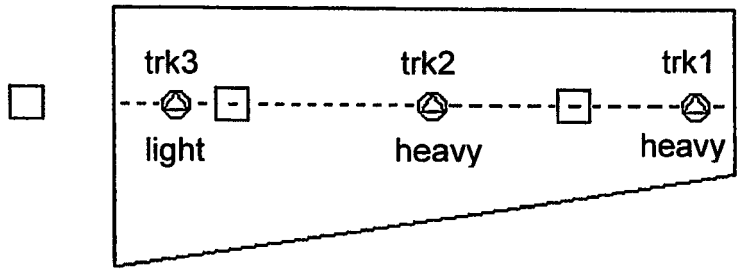


Fig. 12b

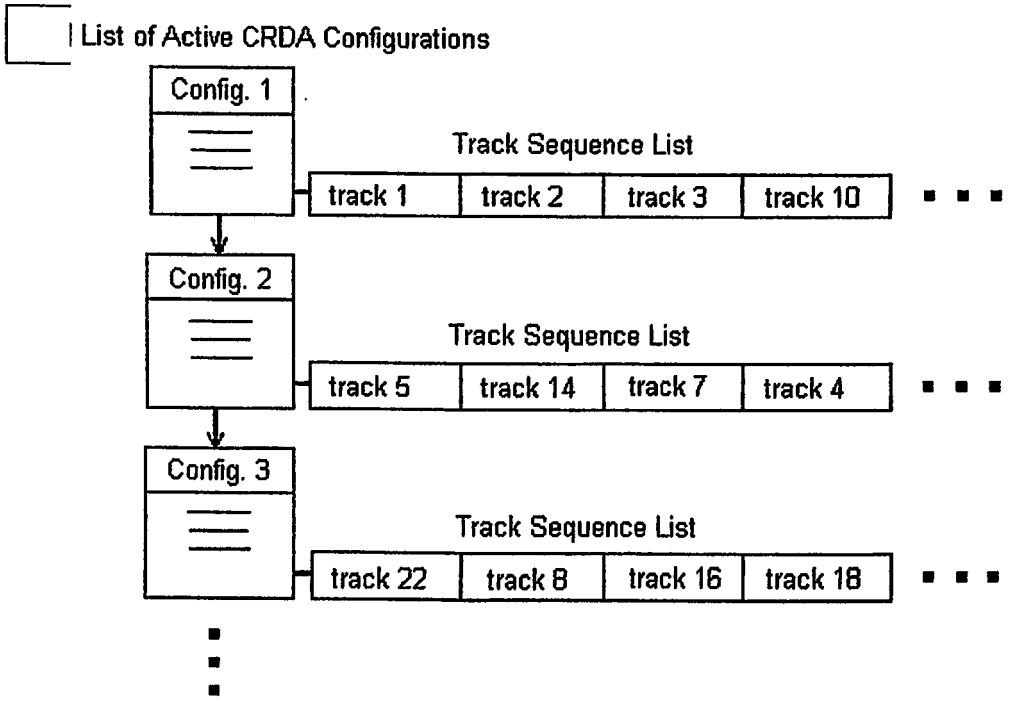


Fig. 13

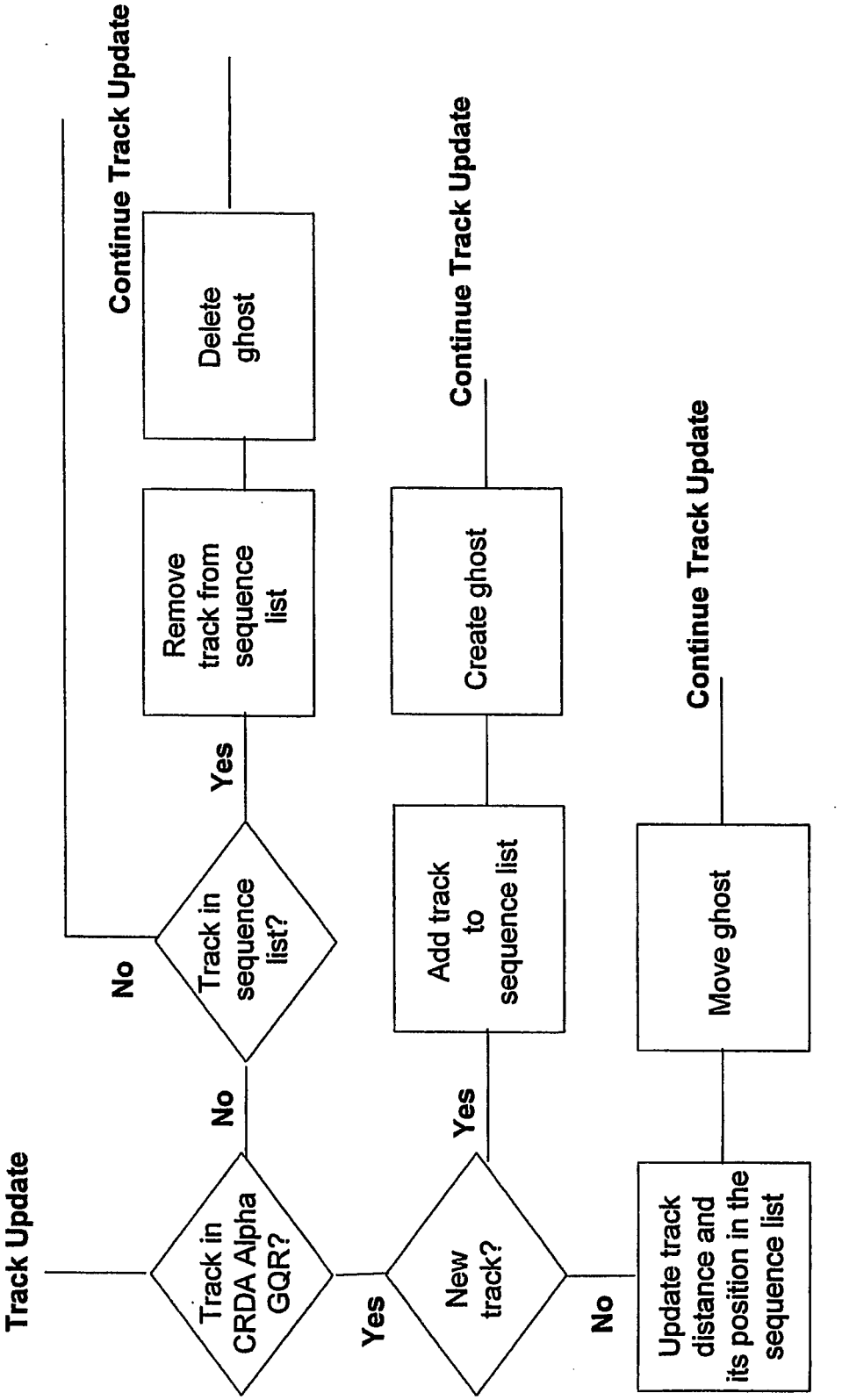

Track Update
Track in CRDA Alpha GQR?
No
Yes
Track in sequence list?
No
Yes
Remove track from sequence list
Delete ghost
Continue Track Update
New track?
Yes
No
Add track to sequence list
Create ghost
Continue Track Update
Update track distance and its position in the sequence list
Move ghost
Continue Track Update

Fig. 14

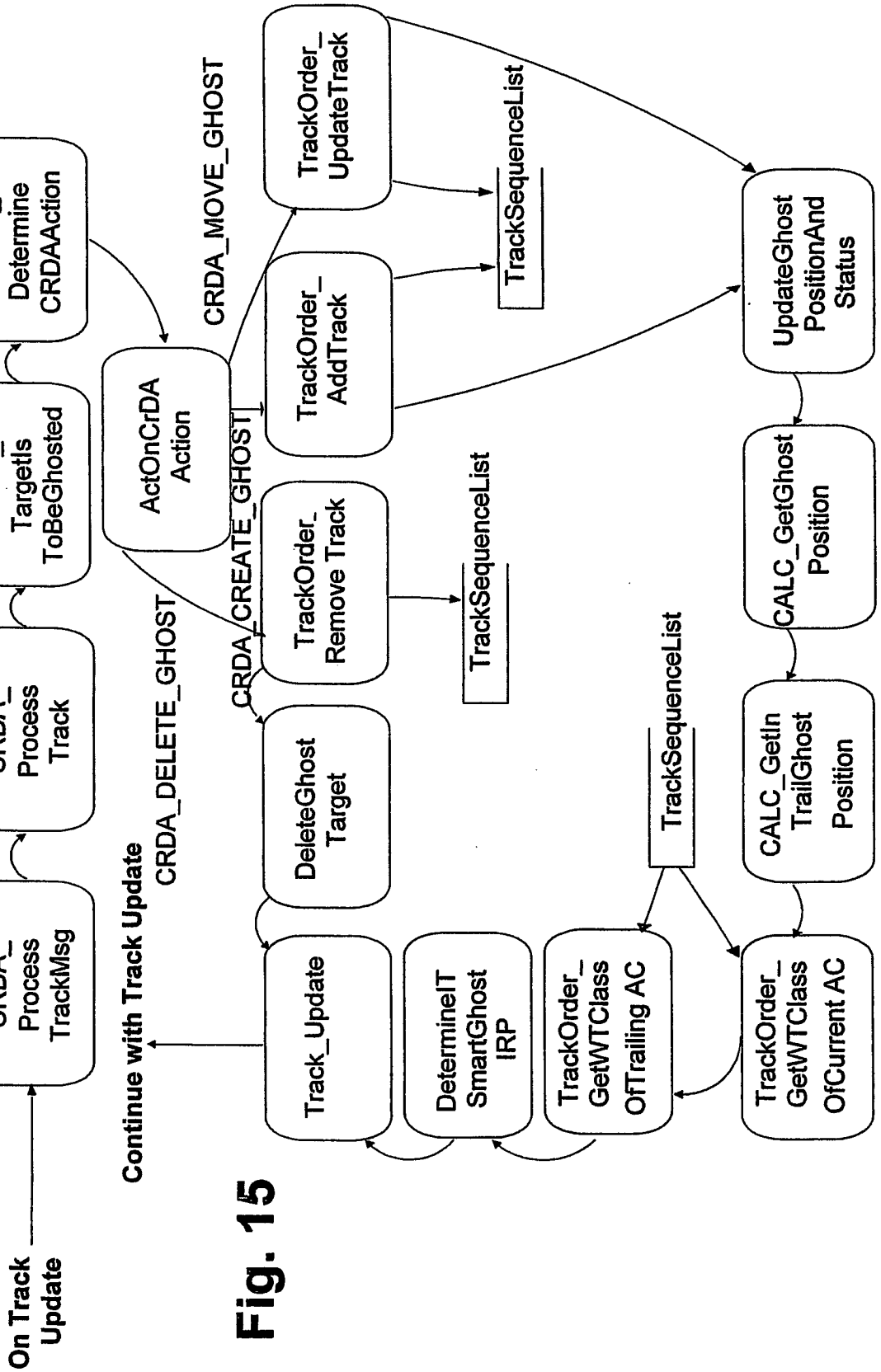

On Track Update
CRDA_ Process TrackMsg
CRDA_ Process Track
CRDA_ TargetIs ToBeGhosted
CRDA_ Determine CRDAAction
Continue with Track Update
Track_Update
CRDA_DELETE_GHOST
DeleteGhost Target
DetermineIT SmartGhost IRP
ActOnCrDA Action
CRDA_CREATE_GHOST
TrackOrder_ Remove Track
TrackSequenceList
TrackOrder_ GetWTClass OfTrailing AC
TrackSequenceList
CRDA_MOVE_GHOST
TrackOrder_ AddTrack
TrackOrder_ UpdateTrack
TrackSequenceList
TrackOrder_ GetWTClass OfCurrent AC
CALC_GetIn TrailGhost Position
CALC_GetGhost Position
UpdateGhost PositionAnd Status

Fig. 15

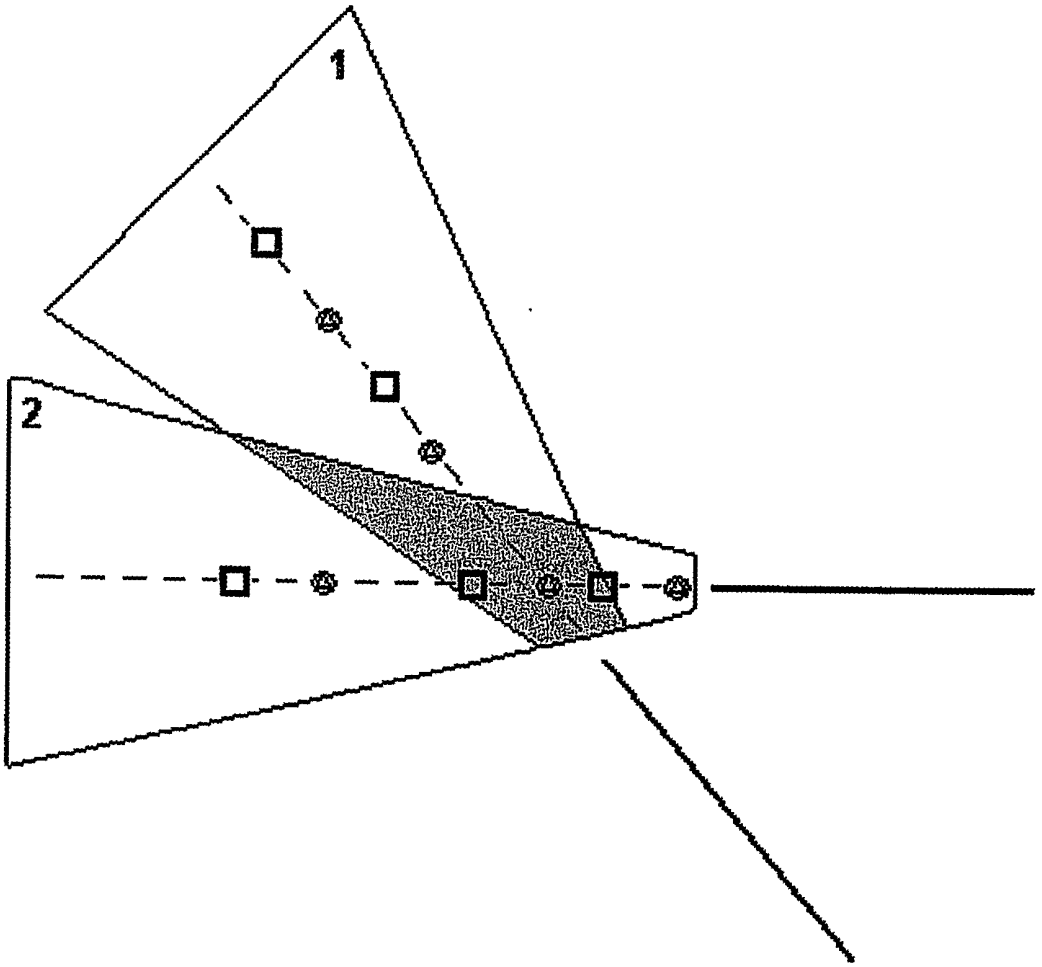


Fig. 16

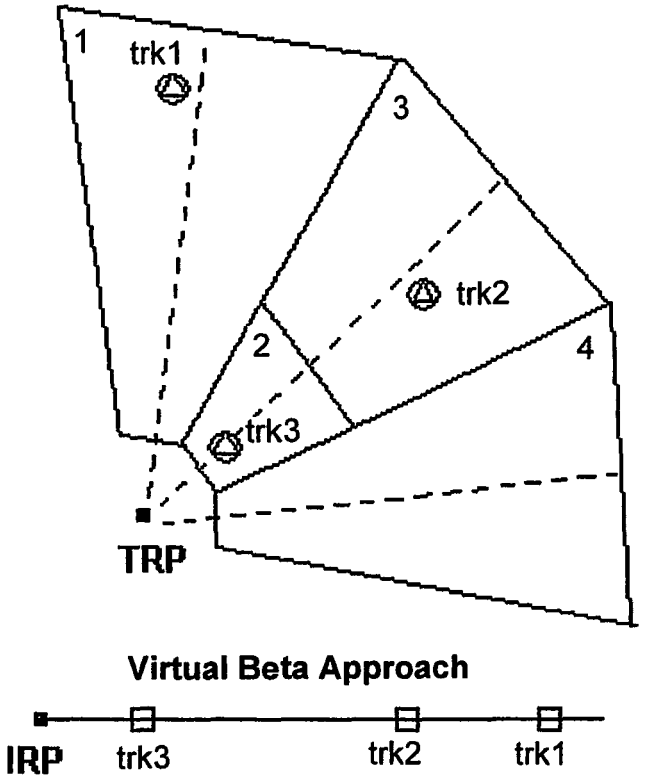


Fig. 17a

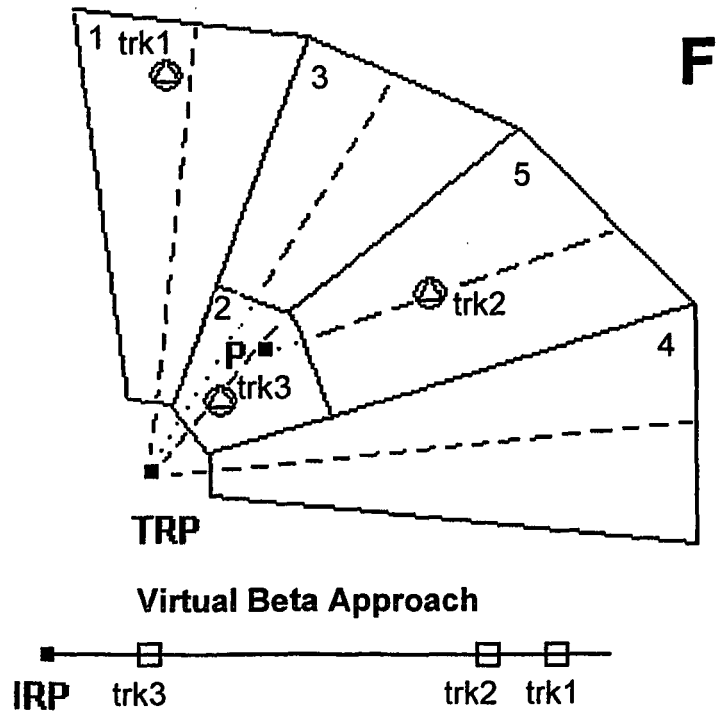


Fig. 17b

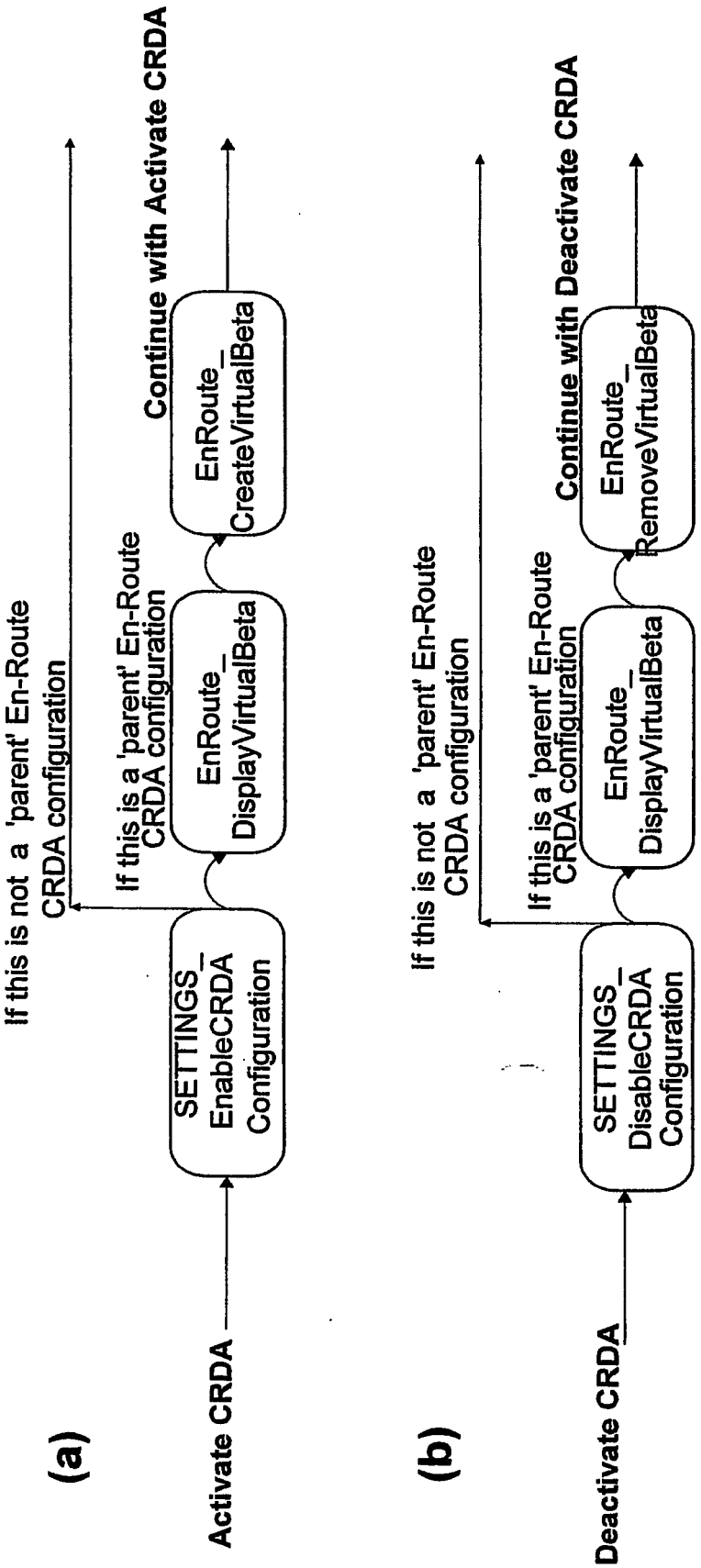

(a)
Activate CRDA
SETTINGS_ EnableCRDA Configuration
If this is a 'parent' En-Route CRDA configuration
EnRoute_ DisplayVirtualBeta
If this is not a 'parent' En-Route CRDA configuration
Continue with Activate CRDA
EnRoute_ CreateVirtualBeta
(b)
Deactivate CRDA
SETTINGS_ DisableCRDA Configuration
If this is a 'parent' En-Route CRDA configuration
EnRoute_ DisplayVirtualBeta
If this is not a 'parent' En-Route CRDA configuration
Continue with Deactivate CRDA
EnRoute_ RemoveVirtualBeta

Fig. 18

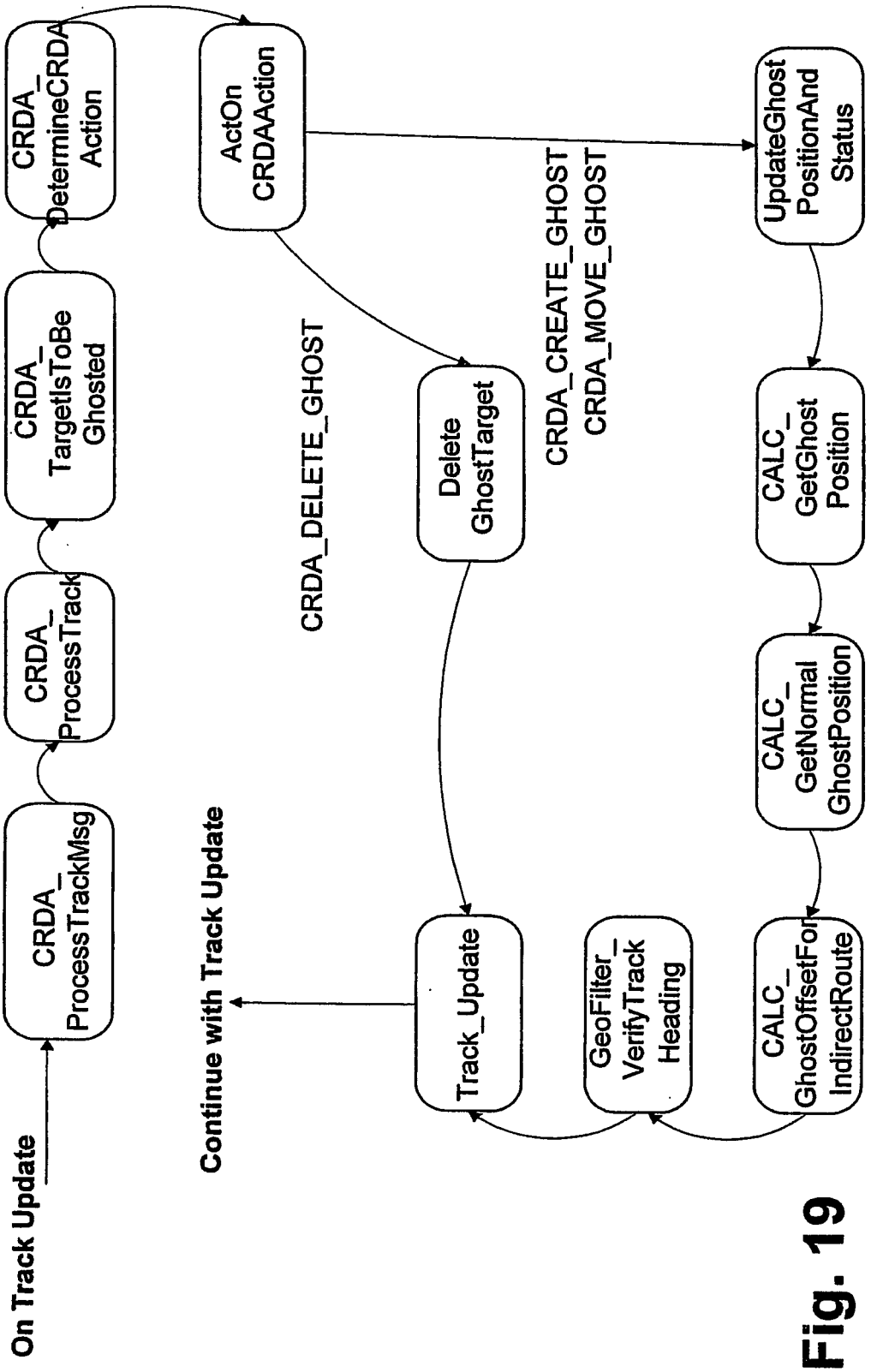

On Track Update
CRDA_ ProcessTrackMsg
CRDA_ ProcessTrack
CRDA_ TargetIsToBe Ghosted
CRDA_ DetermineCRDA Action
ActOn CRDAAction
CRDA_DELETE_GHOST
Delete GhostTarget
Continue with Track Update
Track_Update
CRDA_CREATE_GHOST
CRDA_MOVE_GHOST
GeoFilter_ VerifyTrack Heading
UpdateGhost PositionAnd Status
CALC_ GetGhost Position
CALC_ GetNormal GhostPosition
CALC_ GhostOffsetFor IndirectRoute

Fig. 19

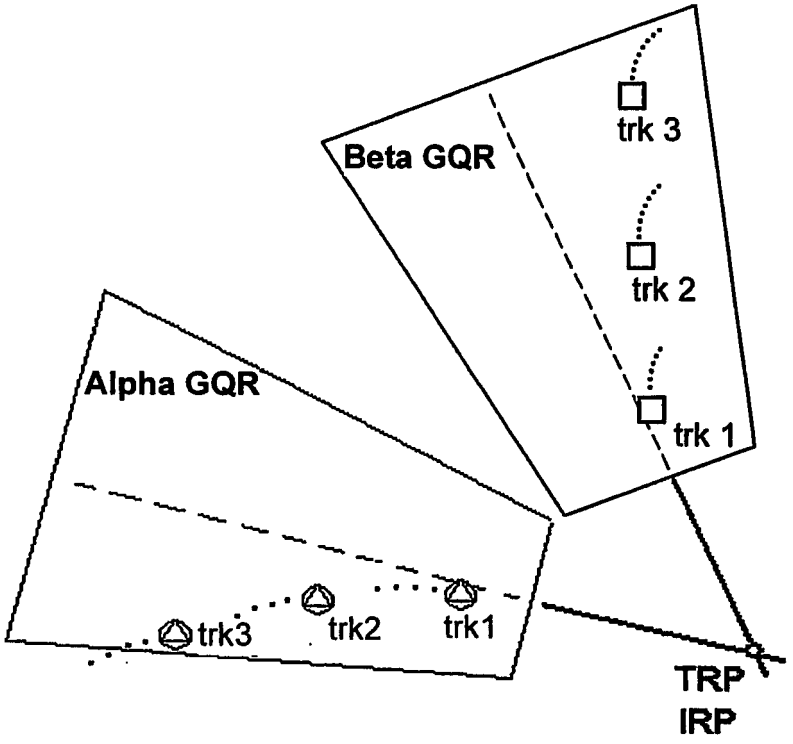

Beta GQR
trk 3
trk 2
trk 1
Alpha GQR
trk3
trk2
trk1
TRP
IRP

Fig. 20

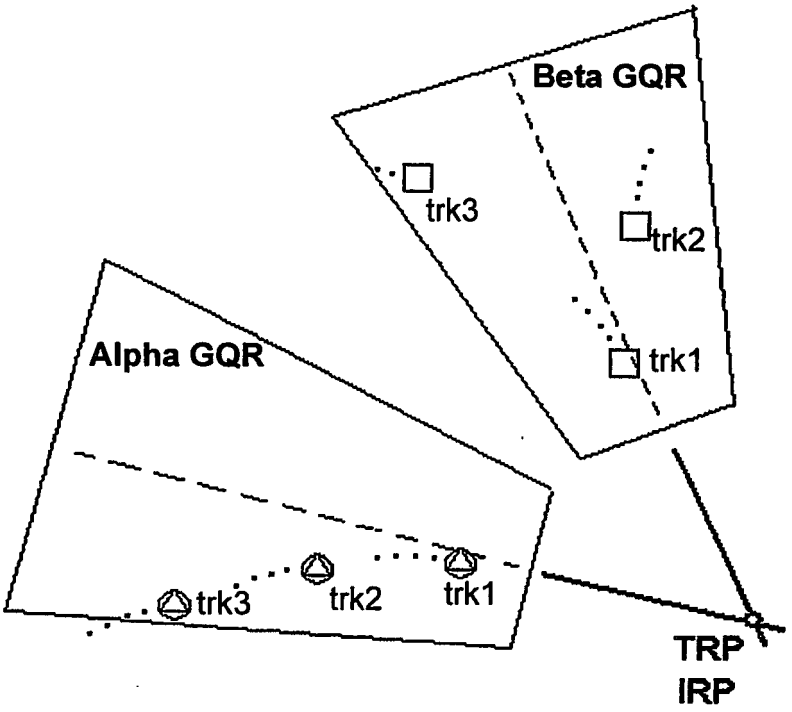

Beta GQR
trk3
trk2
trk1
Alpha GQR
trk3
trk2
trk1
TRP
IRP

Fig. 21

Europäisches Patentamt
European Patent Office
Office européen des brevets

PARTIAL EUROPEAN SEARCH REPORT

Application Number EP 09 15 8253

which under Rule 63 of the European Patent Convention shall be considered, for the purposes of subsequent proceedings, as the European search report

DOCUMENTS CONSIDERED TO BE RELEVANT

Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	CLASSIFICATION OF THE APPLICATION (IPC)
A	MUNDRA A D ET AL: "Capacity enhancements in IMC for converging configurations with down-link of aircraft expected final approach speeds" 20TH DASC. 20TH DIGITAL AVIONICS SYSTEMS CONFERENCE. PROCEEDINGS 14-18 OCT. 2001 DAYTONA BEACH, FL, USA, vol. 2, October 2001 (2001-10), pages 7.E.7-1-7.E.7-13, XP002419905 20th DASC. 20th Digital Avionics Systems Conference (Cat. No.01CH37219) IEEE Piscataway, NJ, USA ISBN: 0-7803-7034-1 * page 5, left-hand column, paragraph 1 * * page 6, right-hand column, paragraph 4 * * page 7 * * page 8, right-hand column, paragraphs 2,3 * * figure 5 * ----- -/--	1-12	INV. G08G5/00 G06F15/00

TECHNICAL FIELDS SEARCHED (IPC)

G08G
G06F
G01C

INCOMPLETE SEARCH

The Search Division considers that the present application, or one or more of its claims, does/do not comply with the EPC to such an extent that a meaningful search into the state of the art cannot be carried out, or can only be carried out partially, for these claims.

Claims searched completely :

Claims searched incompletely :

Claims not searched :

Reason for the limitation of the search:

see sheet C

4

Place of search	Date of completion of the search	Examiner
Munich	11 August 2009	Flores-Jiménez, A

CATEGORY OF CITED DOCUMENTS

X : particularly relevant if taken alone
Y : particularly relevant if combined with another document of the same category
A : technological background
O : non-written disclosure
P : intermediate document

T : theory or principle underlying the invention
E : earlier patent document, but published on, or after the filing date
D : document cited in the application
L : document cited for other reasons

& : member of the same patent family, corresponding document

EPO FORM 1503 03.82 (P04E07)

Europäisches Patentamt
European Patent Office
Office européen des brevets

PARTIAL EUROPEAN SEARCH REPORT

Application Number
EP 09 15 8253

DOCUMENTS CONSIDERED TO BE RELEVANT			CLASSIFICATION OF THE APPLICATION (IPC)
Category	Citation of document with indication, where appropriate, of relevant passages	Relevant to claim	
A	US 2003/200024 A1 (POREDA STANLEY J [US]) 23 October 2003 (2003-10-23) * page 1, paragraph 7 * * page 2, paragraph 13-15 * * page 3, paragraph 35 * * page 5, paragraph 49 * * page 6, paragraphs 53,58 * -----	1-12	
D,A	US 4 890 232 A (MUNDRA ANAND D [US]) 26 December 1989 (1989-12-26) * column 3, line 66 - column 4, line 8 * * the whole document * -----	1-12	
D,A	BURNETT, K, BEASLEY, P., MUNDRA, A.: "Converging Runway Display Aid as a Means to Increase Airport Capacity" ATAC ANNUAL MEETING, October 2000 (2000-10), pages 232-236, XP009078976 * the whole document * -----	1-12	TECHNICAL FIELDS SEARCHED (IPC)

4

EPO FORM 1503 03.82 (P04C10)

Europäisches Patentamt
European Patent Office
Office européen des brevets

INCOMPLETE SEARCH
SHEET C

Application Number
EP 09 15 8253

Claim(s) searched completely:
-

Claim(s) searched incompletely:
1-12

Claim(s) not searched:
-

Reason for the limitation of the search:

The scope covered by the claims does not correspond to that of the description; all embodiments falling within the scope of the claims should be technically enabled by the description but in the present application, the description does not contain the required technical teaching but exclusively one particular embodiment, partially written in pseudo-code for its computer implementation; therefore the search has been limited to the claims as supported, that is, to the scope as exemplarily disclosed in the description.

ANNEX TO THE EUROPEAN SEARCH REPORT ON EUROPEAN PATENT APPLICATION NO. EP 09 15 8253

This annex lists the patent family members relating to the patent documents cited in the above-mentioned European search report. The members are as contained in the European Patent Office EDP file on
The European Patent Office is in no way liable for these particulars which are merely given for the purpose of information.

11-08-2009

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2003200024 A1	23-10-2003	AT 320058 T	15-03-2006
		AU 2003234154 A1	10-11-2003
		CA 2483013 A1	06-11-2003
		DE 60303924 T2	14-12-2006
		EP 1497808 A1	19-01-2005
		JP 4255910 B2	22-04-2009
		JP 2005524157 T	11-08-2005
		WO 03091967 A1	06-11-2003
US 4890232 A	26-12-1989	CA 1299290 C	21-04-1992

EPO FORM P0459

For more details about this annex : see Official Journal of the European Patent Office, No. 12/82

REFERENCES CITED IN THE DESCRIPTION

This list of references cited by the applicant is for the reader's convenience only. It does not form part of the European patent document. Even though great care has been taken in compiling the references, errors or omissions cannot be excluded and the EPO disclaims all liability in this regard.

Patent documents cited in the description

- US 4890232 A, Anand D. Mundra **[0003]**