Design of a PC-Based Command and Control System Using Alpha-Beta-Gamma Digital Tracking Filter

Duraid Fiz'a Mohammad

College of Electronics Engineering
University of Mosul
Mosul-Iraq

Abstract

In this paper, a software model for a Command and Control system is implemented on personal computer. This model can also be used for navigation, or Air Traffic Control (ATC) systems. The hardware required for implementation of this system is also described in this paper. The system assumed to have a fixed position detection source such as the surface search radar. The principle of Track-While-Scan (TWS) that combines the automatic detection of targets and the automatic multi target tracking is used.

The Simplified Alpha-Beta-Gamma (α-β-γ) digital filter is used to perform the algorithm of the automatic tracking. The model main program and all necessary routines are written in C++ language. Successful tests for the implementation of this system are achieved.

Keywords: Command and Control, Track-While-Scan, target, Alpha-Beta-Gamma filter, navigation, ATC

تصميم منظومة قيادة وسيطرة بأعتماد الحاسبة الشخصية وأستخدام مرشح التتبع الرقمي الفا-بيتا-كاما

دريد فزع محمد

كلية هندسة الألكترونيات
جامعة الموصل

الخلاصة

تم في هذا البحث تصميم وفحص عمل أنموذج برمجي لمنظومة قيادة وسسيطرة باستخدام الحاسبة الشخصية. وهذا النموذج يمكن تطبيقه ايضا" لمنظومات الملاحة أو السيطرة الجوية. كذلك تم وصف المتطلبات المادية لتنقيذ مثل هكذا منظومة. تم تصميم هذه المنظومة لتعمل على كشف الأهداف بأستخدام الرادار الأرضي الثابت ومبدأ التتبع خلال عملية المسح والذي يجمع بين مبدأ الكشف والمتابعة الطوعيان لأكثر من هدف في آن واحد.

تم في هذا البحث استخدام المرشح الرقمي الفا-بيتا-كاما المبسط لتنفيذ خوارزميات التتبع الطوعي، واعتماد لغة C++ في بناء هذة المنظومة، علما" انه قد تم اختبار عمل هذه المنظومة بنجاح.

Received: 3 – 3- 2011 **Accepted: 4 – 7 - 2011**

1. Introduction

Command and Control systems have the facilities to perform multiple functions, including surface, subsurface, and airspace targets detection; tracking; surveillance; and identification [1, 2]. These systems are widely used around the world for civil or military applications. They are extremely similar to navigation systems, or Air Traffic Control (ATC) systems, and most of the time they considered as the main part of the C3I systems. These systems commonly used sophisticated high-speed digital computers and special software programs, which make them very expensive, and the end user of these systems needs the supplier for any software development.

Most of the Command and Control systems depend on a high scan rate (10-20 rpm) commercial two-dimensional (2D) or three-dimensional (3D) surface radar for the detections of targets. The detected targets are discriminated and fed to digital computer for the process of tracking; surveillance; and indemnification. Such detection sources are suitable for the Command and Control systems to perform the principle of Multi Target Tracking (MTT), which means that more than one target can be tracked simultaneously [3]. Automatic tracking for targets is performed periodically as their data is updated from the detection source, which is referred as the Track-While-Scan (TWS) concept [4].

In this paper, we try to introduce a low cost effective, reliable and flexible software Command and Control system. This was achieved by using a personal computer (PC) in the system, and writing the software model by the popular C++ programming language.

The suggested Command and Control system in this paper work with the same concepts of MTT and TWS mentioned before. The system consists of the radar system simulator, which used as the detection source replacing the surface radar for the experimental tests. This simulator has the ability of generating fixed or moving targets of different speeds and directions, with different scan rates. To convert the target’s echo signal received from search radar or the radar system simulator to the digital (binary) form, target extractor is used, and the data are fed to the PC via a high speed rate data channel. The main purpose of the PC is to receive target binary data; process it for the automatic detection; initiation of targets to be tracked; update data for previously tracked targets; interface line commands to system; and finally viewing the radar data and the associated Command and Control system requirements on a graphics display unit.

Tracking algorithm is accomplished in this paper using the simplified Alpha-Beta-Gamma (α-β-γ) digital tracking filter, which is capable of tracking of accelerated targets. The filter computes the predicted target position and it’s velocity depending on the previously calculated data and the measured position. This filter is suitable to be used in Command and Control system working in the Track-While-Scan (TWS) concept [5, 6].

In this paper, the detailed hardware required for the suggested Command and Control system is discussed in section 2, while section 3 of this paper discuss the detailed software routines required for running the software model on the PC. The principles of operation and the detailed equations for the α-β-γ digital filter applied for tracking algorithms is given in section 4. The results obtained from the theoretical test of the α-β-γ digital filter, and a sample layout of the model software run on the graphics display unit are given in section 5. Finally section 6 concludes this paper.

2. System Hardware Requirements

The general block diagram of the Command and Control system is shown in Fig (1). As shown in figure, in addition to personal computer required for the software model implementation of the suggested Command and Control system, special hardware equipments are added to integrate the software model with the whole Command and Control system, include the followings:

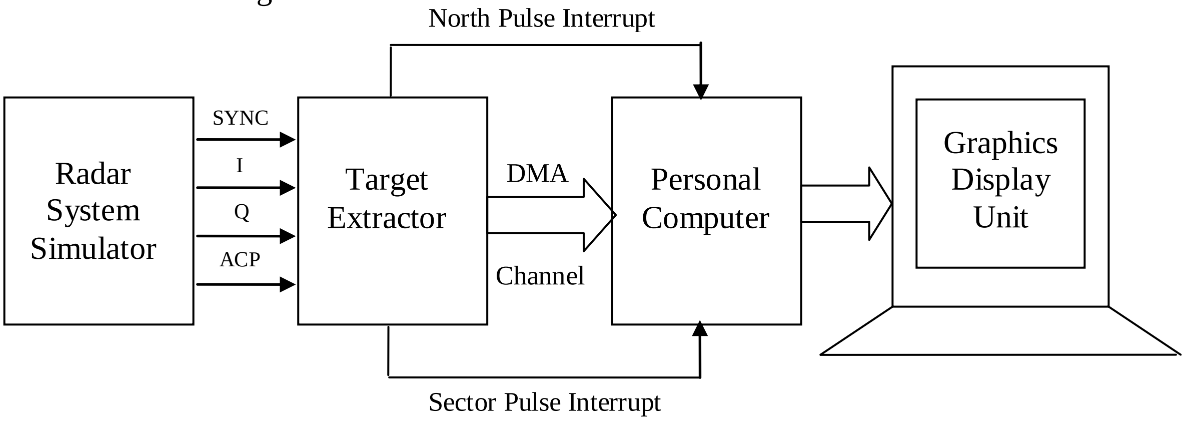


Fig (1) General Block Diagram of the Command and Control system

2.1. Radar System Simulator

A standalone personal computer with an AT bus interface card radar system simulator driven by a special program is used in this model to replace the search radar for experimental implementation and test of the Command and Control system model. The radar system simulator has the ability and flexibility to generate the signals of different radar types, such as the reference synchronous radar signal (SYNC), with the I and Q analog signals for representing of target range and position, Angle Pulse signal (ACP) of (4096) square pulses per radar antenna turn to represent target angle, and North Pulse (NP) signal one pulse per antenna turn for the indication that the radar antenna is in the real north.

The system is capable of initiating up to (12) targets of maximum range resolution of (30) meters; maximum antenna scan rate up to (60) rpm; any target speed; and adjacent two targets within angle greater than (0.5°) for range greater than (300) meters [7].

2.2. Radar Targets Extractor

An AT bus interface radar targets extractor card plugged into the same PC is used to convert targets echo data received from search radar or radar simulator to the binary form [8]. The extractor is equipped with buffer memory organized into blocks each of (8) bytes. The code in the first (4) bits represent start or end block of a target, followed by (12) bits to represent the target relative range, (12) bits for number of Angle Pulses (ACP) elapsed from last north pulse, and (12) bits representing relative target elevation when extractor is connected to 3D detection source. Thus, true targets will have two blocks of data, start block holds the start ACP pulses, and end block holds the end ACP pulses for the target. The two blocks should have the same range cells.

The north pulse is applied directly to one of the user interrupts of the AT bus, and a programmable counter is used to generate a pulse for each certain number of ACP pulses. This pulse is also applied directly to another interrupt of the AT bus which is referred as sector pulse interrupt. The sector pulse is used for dividing the radar turns into number of sectors; ensure complete data transfer to computer memory; and to get a reasonable smooth movement of the cursor (radar trace) on the graphics display unit. In our model the extractor is programmed to generate (32) sector pulse for each complete radar turn.

To ensure a very high data rate transfer form the targets extractor buffer to the PC memory, the Direct Memory Access (DMA) technique is used to program one of the user channels for this purpose.

3. Model Software Flow Diagram

The general software flow diagram for the Command and Control system model is shown in Fig (2) below. The diagram consists of several routines driven by a main program. Some of these routines are executed sequentially, and others are controlled by special interrupts to be executed whenever main program needs them.

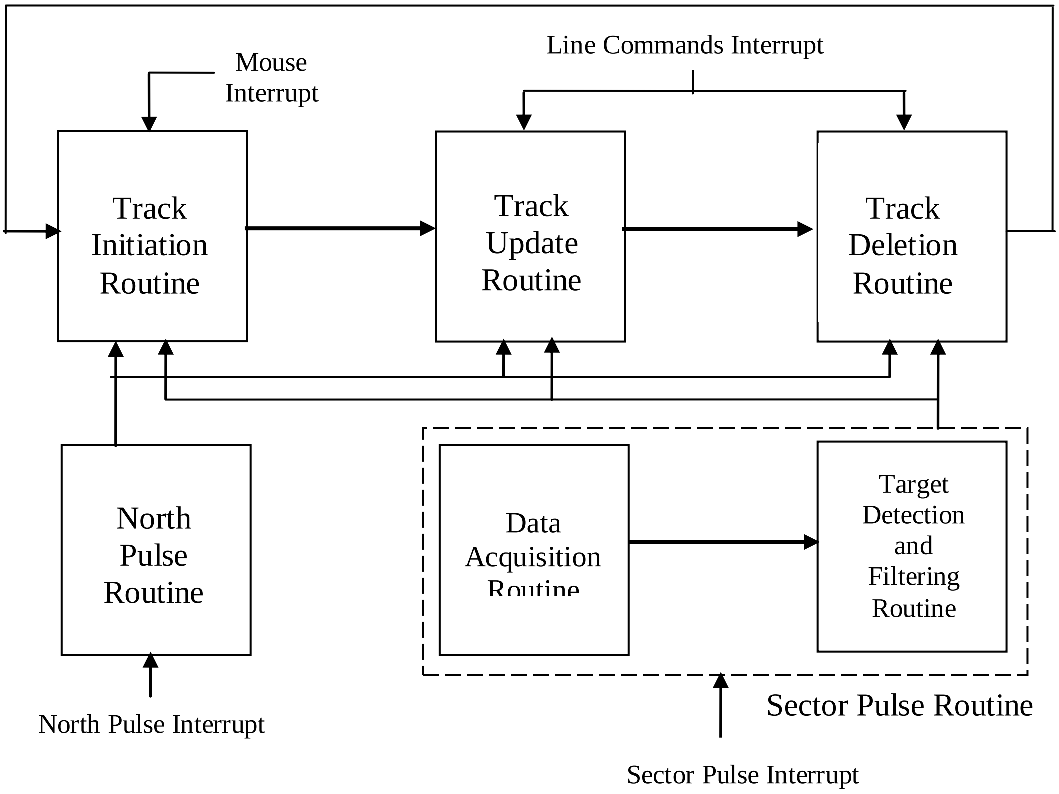


Fig (2) General Software Flow Diagram of the Command and Control System

3.1. Sector Pulse Routine

This routine is executed whenever the sector pulse interrupt goes active. It consists mainly of two routines: the data acquisition routine, which is related to the programming and re-initialization of the (DMA) channel to manage the data transfer from the targets extractor buffer to PC memory [9], and the target detection and filtering routine, which is related to the investigation of data transferred. Target is detected when the routine receive two blocks of data (start and end) of the same range and elevation data cells. Once target is detected, this routine computes its position in polar form depending on blocks data which contains relative range; relative azimuth (azimuth is calculated by taking the mid of the ACP counts in start and end blocks); and the relative elevation if the targets extractor is connected to a 3D detection source.

Targets data obtained in polar form are converted to the Cartesian form to simplify the tracking algorithm using the following formulas [10]:

$$x = R \cos \varepsilon \cos \Phi \quad (1)$$

$$y = R \cos \varepsilon \sin \Phi \quad (2)$$

$$z = R \sin \varepsilon \quad (3)$$

Where R is the computed target range in kilometers, ε is the target elevation in degrees, and Φ is the target azimuth in degrees.

As the target is detected, it will be viewed (plotted) on the graphics display unit, at end of this routine the cursor of the graphics display unit will be forwarded by the exact sector angle erasing all the previous plots. All targets detected in this routine will be stored in array called the targets array.

When the Command and Control system model is implemented to real search radar, due to noisy the targets extractor may generate false blocks of targets data, targets have start block but without end block, and vise versa; too short targets; or too long targets. The filter routine will erase all these blocks.

3.2. North Pulse Routine

This routine is executed when the radar antenna is exactly directed to the real north, i.e. when the NP interrupt is set active. This is a simple routine for the indication of beginning of new radar turn. In this routine all the counters and flags related to targets used in main program and routines are nulled. The routine also overcome any error occurs in the sequence of sectors, and moves the radar cursor in the graphics display unit to the north position.

3.3. Track Initiation Routine

This routine is used to mark the desired targets on the graphics display unit to be tracked. In this paper the targets set to the automatic tracking mode are referred to as tracks. In Command and Control systems there are two concepts of initiation tracks: the automatic initiation which means that all targets detected will be treated and served as tracks, and the manual initiation described in this model in which the operator or the system user has the ability to select certain targets to be tracked.

Track Initiation routine is activated by a single click of the mouse left button on the target viewed in the graphics display unit. The first job in this routine is to reject multiple mouse clicks on same point on graphics display unit, the pressed relative position in converted to Cartesian components. A designation gate around position is opened, the routine will check all targets positions in the targets arrays within this gate. If no target is found, order is neglected but if the routine got a single target in this gate, and the target is not previously marked as a track, then it will be considered as the desired target. Finding more than one target in the gate, the routine will choose the Nearest Neighbor target to the gate center as the desired target [4, 11]. The routine will replace the order position with that got from the targets array and an object is created and filled with initial data. This data will represent the initialization to the α-β-γ filter and the object is considered as a first stage track and will be appended to the tracks array in the main C++ program [12].

3.4. Track Update Routine

Track update routine is sequentially executed throughout the model program. It is associated with the updating of tracks data and their plots on the graphics display unit. Automatic tracking algorithm using α-β-γ digital filter is performed on all tracks. Tracks at first stage are updated by opening a tracking gate around its predicted position, computed from previous radar turn; check for the existing of single, or Nearest Neighbor target in the tracking gate; get measured position; compute the filtered and predicted position of the track using the α-β-γ filter equations as will be described later; compute the track velocity, and updates object data to the tracks array. When the track passes this stage it will be marked as second stage track. If no associated target is found in the tracking gate, the target object will be deleted from the tracks array [4, 11, 13].

Updating second stage tracks is similar to the algorithm applied for first stage tracks except that target velocity will be considered as a factor in calculating the predicted position of the track. Tracks pass this stage will be considered as final tracks, and they will be discriminated on the graphics display unit by drawing a red circle around their predicted positions, with a short line to represent heading (direction) of the track. Each final track in the model will get a different numeric integer number, and track data (number, range, azimuth, and elevation) will be displayed on numeric table on the graphics display unit.

Updating final tracks in this routine is similar to the algorithm discussed before, the difference is in the implementation of the α-β-γ filter equations for these tracks as will be discussed later. In this update procedure if there is no existence for targets in the tracking gate, the previous predicted position of the track will be taken as the measured position and updating is continued.

Fig (3a) shows the Nearest Neighbor target theorem for the designation or tracking gates, Fig (3b) shows the theorem of having a common Nearest Neighbor target for two gates of tracks.

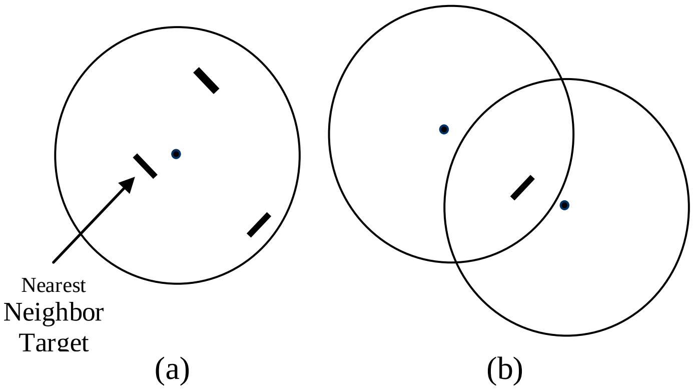


Fig (3) The Nearest Neighbor target theorem

3.5. Track Deletion Routine

This routine manages the deletion process of the marked tracks, including deleting of object from the tracks array, and erasing the related tracks signs and data in numeric table on the graphics display unit. Deleting tracks in first and second stages is accomplished automatically as has been discussed before. Final tracks are deleted automatically if the associated track data disappeared from targets array (track has empty tracking gate) for certain number of radar turns. When the track data is disappeared, the model will prompt the operator by replacing the red colored circle around track with the yellow color. The number of radar turns for deleting tracks depends on tactical specification for the Command and Control system, and it is usually taken between 5 and 10. Final tracks can also be deleted manually by a special operator command throughout the line commands routine.

3.6. Line Commands Routine

In order to integrate the operation of the Command and Control system, the model is interfaced by the line commands routine, which enables the operator to manage the operation of the system [14]. These commands are executed on operator request by typing them using the PC keyboard, to interrupt the model main program. These commands are viewed on the command field box, and the result of executing these commands are displayed on the message box field on the graphics display unit.

This routine may contain several commands, depending on the requirements of the end user of the system. Examples of these commands are listed below:

Track Delete command (***Delete n***) where n is the track number to delete; this command will delete the track data object and update the graphics display unit accordingly.
Track Correct command (***Correct n***) which enables the operator to position the predicted position of certain track, track data and its plot will be updated immediately. Track new position is set by mouse click on the graphics display unit.

Track Elevation Set command (***Elev n = xxxx***), enables the operator to correct or set elevation of a certain track, this command is often used to set the elevation of tracks when system is connected to 2D detection source, and the elevation is got by another source.
Send Track command (***Send n***), to sent data of a certain track through available communication port.
Radar Range Change command (***RDR xxxx***), to set a new radar range to match the range of the detection source. Executing this command will refresh (redraw) the graphics display unit to get targets and tracks on their new relative positions according to the new radar range.

4. The α-β-γ Digital Tracking Filter:

The Alpha-Beta-Gamma (α-β-γ) filter is a simplified form of sampled data observer for estimation, data smoothing and control applications. It is closely related to Kalman filter with its principal advantage that it does not require a detailed system model. This type of filter is quite suitable for target tracking in Control and Command systems working in the Track-While-Scan concept. The filter is able to track accelerating targets, besides the prediction of position and velocity. Final tracks are updated in this model using the following equations [15, 16]:

$$x_P(k+1) = x_s(k) + T.v_s(k) + \frac{1}{2}T^2.a_s(k) \quad (4)$$

$$\mathrm{v_p(k+1) = v_s(k) + T.a_s(k)} \quad (5)$$

$$\mathrm{x_s(k) = x_p(k)} + \alpha\mathrm{(x_0(k) - x_p(k))} \quad (6)$$

$$\mathrm{v_s(k) = v_p(k)} + \frac{\beta}{\mathrm{T}}\mathrm{(x_0(k) - x_p(k))} \quad (7)$$

$$\mathrm{a_s(k) = a_s(k-1)} + \frac{\gamma}{\mathrm{2T^2}}\mathrm{(x_0(k) - x_p(k))} \quad (8)$$

where:

x_P (k+1) is the predicted target position for the next radar turn
v_P (k+1) is the predicted target velocity for the next radar turn
x_S (k) is the filtered (smoothed) target position in current radar turn
v_S (k) is the filtered (smoothed) target velocity in current radar turn
x_P (k) is the predicted target position computed from previous radar turn
v_P (k) is the predicted target velocity computed from previous radar turn
x_o (k) is the measured target position for current radar turn
a_S (k) is target acceleration in current radar turn
a_S (k-1) is the target acceleration for previous radar turn
and

T is the radar time interval (time required by radar antenna to complete a single turn)

The α-β-γ filter requires two steps of initialization, which are referred to tracks in first and second stages. In first stage tracks the predicted position is set as the measured position, while velocity and acceleration are nulled. For second stage tracks predicted target position is calculated using the relation of velocity, while acceleration is calculated as track pass to the final stage and predicted target data is computed using the previous equations.

Typically, values of α, β, and γ are adjusted experimentally. In general large α and β gains tend to produce faster response for tracking transient changing, while small α and β gains reduce the level of noise in the state estimations. Usually value of α is taken between 0 and 1, in this model α is taken (0.7), for a good balance between accurate tracking and noise reduction, values for β and γ are calculated using the following equations [10, 17]:

$$\beta = \frac{\alpha^2}{2-\alpha} \tag{9}$$

Or $\beta = 2(2-\alpha) - 4\sqrt{1-\alpha}$ (10)

and $\gamma = \frac{\beta^2}{2\alpha}$ (11)

5. Results

The system have been tested theoretically and practically, theoretical test for the α-β-γ digital filter, whereas practical test applied to system by running the model software on PC with the system hardware and recording results on the graphics display unit.

5.1. The α-β-γ filter

The α-β-γ filter is tested by special programs, in which filter equations are implemented to compute the predicted position of a suggested moving target with constant velocity, and another with constant acceleration. The plot of actual and predicted positions versus time of the mentioned targets is shown in Fig (4a) and Fig (4b) respectively.

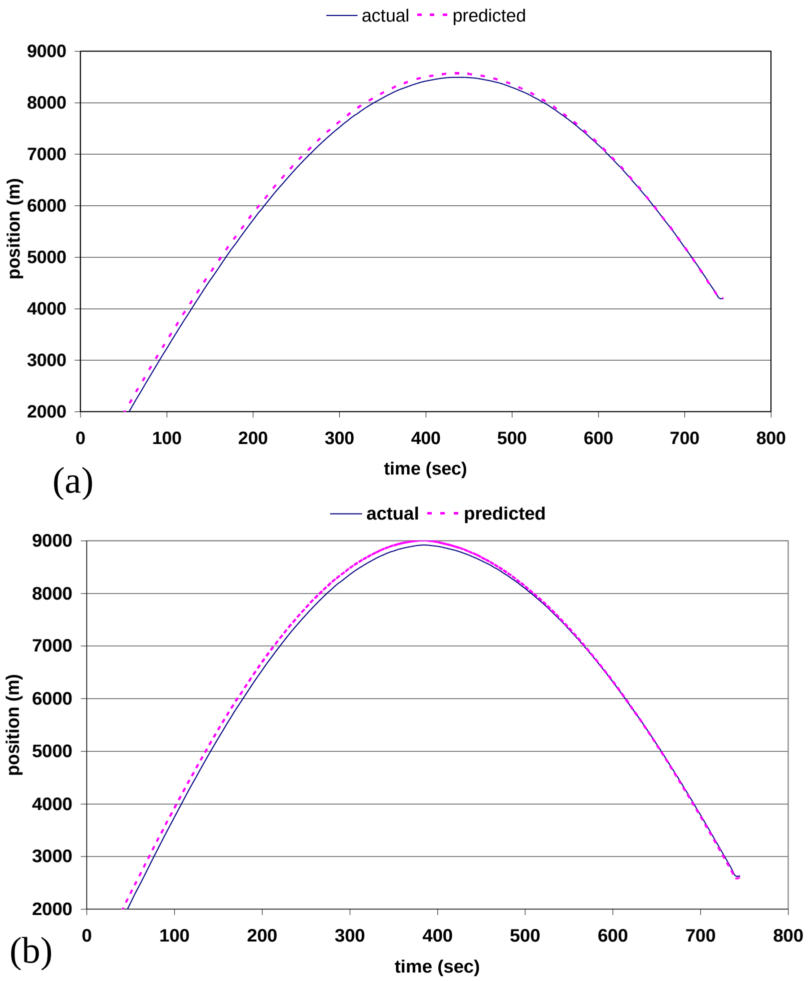


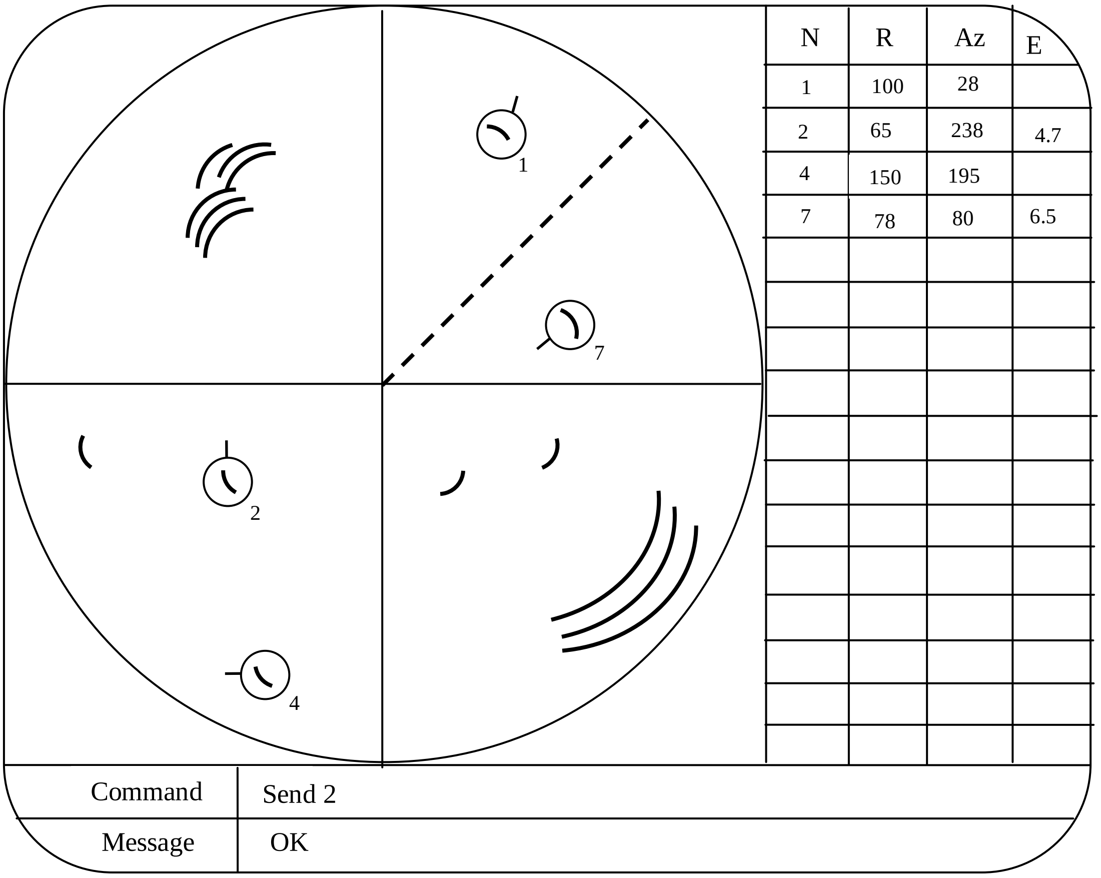
Fig (4) Plot of actual and predicted track position versus time

5.2. The Graphics Display Unit:

The graphics display unit is an important integrated part of the Command and Control system. It gives the operator the ability of monitoring targets detected by surface radar, initiation and follow up marked tracks, and managing of the system. Commercial computer graphics screen color is a good and low cost choice to be as the graphics display unit for the Command and Control system. In this system, 17"size screen with high resolution (800x600) pixels graphics driver is used, certainly larger screen size with higher resolution graphics driver will be better.

The layout of the graphics display has a circular part with moving cursor (trace) to simulate the radar screen. Moving cursor will indicate the instantaneous position of the radar antenna, this region is used to view all targets detected by the extractor exactly at their relative positions. Tracks are marked by red circles with indicators to their directions, besides the numeric number given to each track, which is important for the operator for monitoring target data on numeric table, and to deal with the track by its number through the line commands.

The screen displays a numeric table to view all tracks data, which contains fields for track number (N); track range in kilometers (R); track azimuth in degrees (Az); and track elevation in kilometers (E). The bottom region of screen is used to view commands entered by the operator, and the message field is related to result of executing line commands. Fig (5) shows a sample layout of the graphics display unit.



N	R	Az	E
1	100	28	
2	65	238	4.7
4	150	195	
7	78	80	6.5

Command	Send 2
Message	OK

Fig (5) Sample Layout of the Graphics Display Unit

6. Conclusion

In this paper, the detailed hardware and software needed for the Command and Control system is discussed. This model has the advantage of low cost since it depends on personal computer, which is widely used and available in the markets. Software model needed for the system is written using a very popular programming language (the C++ language), and it is organized into several routines which make the model easily adaptive in future to any extra requirements. The model has been tested experimentally for different static and moving targets with different speeds, the system is capable of tracking (16) targets at the same time. The model was also tested tactically by real 2D search radar and good results are obtained.

From the experimental and tactical test of the model, it is found that the α-β-γ filter is quite suitable for the operation of such models for the tracking purposes of moving and constant accelerating targets, but this model can easily be reformulated to use the Kalman filter for tracking purposes.

Although this paper does not contain any listing for main program and routines, but it is available to any researcher having the interest in this field to contact the author directly.

References

1. Luqi V. Berzin, M. Shing, N. Nada, and C. Eagle, "Software Evolution Approach for the Development of Command and Control Systems", Computer Science Department, Naval Postgraduate School, Monterey, CA93943, supported by the U.S. Army Research Office, Contract Number 35037-MA and 40473-MA.
2. Richard Karlson, "Simulation Based Methods for Target Tracking", Thesis No. 930, Division of Automatic Control and Communication Systems, Dept. of Elect. Eng., Sweden, 2002.
3. D. J. Wilkin, I. Harrison, M. S. Woolfson, "Target Tracking Algorithms for Phased Array Radar, IEE Proceeding-F, Vol. 138, No.3, June, 1991, pp. 255-262.
4. T. R. Bendict, G. W. Bordner, "Synthesis of an Optimal Set of Radar Track-While-Scan Smoothing Equations", IRE Transaction on Automatic Control, Vol.7, No.4, July, 1962, pp. 27-32.
5. W. D. Blair, "Fixed Gain Two Stage Estimation for Tracking Maneuvering Targets", IEEE Transaction on Aerospace and Electronic Systems, Vol.29, No.3, July 1993, pp. 1004-1014.
6. Terrence L. Ogle, William D. Blair, "Fixed-Lag Alpha-Beta Filter for Target Trajectory Smoothing", IEEE Transaction on Aerospace and Electronic Systems, Vol.40, No.4, Oct. 2004, pp. 1417-1421.
7. M. A. Al-Zubaidy, "Real Time Adaptive MTI Using DSP", Ph. D. Thesis, Dept of Elect. Eng., College of Eng., University of Mosul, Iraq, 2006.
8. Bassem R. Mahafza, "Radar Simulation Analysis and Design Using Matlab", Chapman Hall/CRC, 2000.
9. Michael Tischer, "PC Intern System Programming, The Encyclopedia of Programming Know How", Abacus, USA, 1995.
10. Dirk Tenne, Tarunraj Singh, "Characterizing Performance of α-β-γ Filters", IEEE Transaction on Aerospace and Electronic Systems, Vol.38, No.3, July 2002, pp. 1072-1087.
11. Darko Musicki, Sofia Suvoruva, "Target Track Initiation Comparison and Optimization", Center of Expertise in Networked Decision and Sensor Systems, Citeseer, 2009.
12. Herbert Schildt, "C++ The Complete Reference", 3rd Edition, Osborne McGraw-Hill, 1998.
13. Darko Musicki, Mark R. Morlande, "Gate Volume Estimation for Target Tracking", Research Center for Sensor and Information Processing, Citeseer, 2009.

14. Shahid Manzoor, Werner Cuesters, and Barry Smith, “Referent Tracking for Command and Control Messaging Systems”, Center of Excellence in Bioinformatics & Life Sciences, University at Buffalo, Citeseer, 2010.

15. John H. Painter, David Kerstetter, and Steve Jowers, “Reconciling Steady State Kalman and Alpha-Beta Filter Design”, IEEE Transaction on Aerospace and Electronic Systems, Vol.26, No.6, Nov. 1990, pp. 986-991.

16. John E. Gray, William Murray, “A Derivation of Analytical Expression for the Tracking Index for the Alpha-Beta-Gamma Filter”, IEEE Transaction on Aerospace and Electronic Systems, Vol. 29, No. 3, July 1993, pp. 1064-1065.

17. Dirk Teme, Tarunraj Singh, “Optimal Design of α-β-γ Filters”, ONR/SPAWAR Systems, Center Grant Number 150-1284A, Citeseer, 2009.

The work was carried out at the college of Electronics Engineering. University of Mosul