TRISS ERROR ANALYSIS

Part 1 of 2

By

Luis E. Alvarado

OUTLINE

1. INTRODUCTION

1.1 Scope

This document describes the analysis conducted to evaluate the susceptibility of the Tracking Infrared Scoring System (TRISS) to site measurement errors for different TRISS configurations. Four different cases were evaluated:

(1) Present TRISS configuration. The distance between Camera 1 (C1) and Camera 2 (C2) and the angles between the cameras and the target are measured using extremely precise surveying equipment, but imprecise UTM positions for Cameras 1 and 2 are used to estimate the angle between C1 and C2 and to determine true North.

(2) The cameras and target UTM positions are measured using a hand held GPS device that normally provides position with a Circular Error of Probability (CEP) of approximately 15 feet.

(3) The distance between camera 1 and camera 2 and the angles between the cameras and target are measured using imprecise surveying equipment (i.e. using the cameras pan/tilt device to measure angles).

(4) Ideal TRISS Configuration. The positions of C1 and C2 and the angles between the cameras and the target are measured using precise surveying equipment.

1.2 Background

TRISS is a ground based scoring system that uses two cameras with infrared lenses to determine the distance between a pre-selected target and the position where the deployed warhead hits the ground. TRISS also determines the scoring angle with respect to the aircraft that deployed the bomb (i.e. ingress angle).

2. CONCLUSION

The analysis shows that the accuracy of the scoring error is totally dependent in the accuracy of the internal angles between the cameras and the target used for scoring and the accuracy of the position of Cameras 1 and 2. The analysis also shows that the absolute position of C1 and C2 is not necessarily needed as long as the distance between C1 and C2 is accurately known and TRISS "knows" where true North is.

3. ANALYSIS ASSUMPTIONS

The analysis assumes that most of the TRISS errors are generated because the exact position of the cameras are not available or the angles between camera 1 and camera 2

and the target are not known with the desired precision. This analysis makes the reasonable assumption that the scoring errors generated because of the pixel camera resolution are negligible compared to the errors generated because of measurement errors with the cameras positions and inaccurate internal angles measurements.

4. TRISS MATHEMATICAL MODEL

The analysis was conducted using a Monte-Carlo simulation model of TRISS. This model was developed using the MATLAB program. The MATLAB program is a 4th generation language that provides the user with the necessary tools to easily generate math models of complex physical systems. Appendix B includes a list of the program.

The TRISS mathematical model assumes that the target is co-centric to cameras 1 and 2. Camera 1 position is fixed and camera 2 rotates around the target. The TRISS model computes the estimated target position error every time C2 is rotated one degree around the target. The user has the capability to modify C1 and C2 starting positions as well as the radius between C1 and the target. The user also has the capability to vary camera position and angular measurement errors.

4.1 Case 1 Error Model

In Case 1 the mathematical model assumes that the angles between C1 and C2 and the target are precisely known. The model also assumes that the distance between the C1 and C2 is also exactly known. The model assumes that an imprecise GPS device is used to measure the position of C1 and C2 in the UTM coordinate system. A standard non-differential, C/A code, GPS will provide and accuracy of 15 feet when Selective Availability (SA) is turned off. The model also assumes that a mathematical triangulation algorithm is used to estimate the target position. It should be noted that UTM position data for C1 and C2 are required to correlate the C1, C2 and Target internal angles to a true North-East coordinate system (i.e. UTM). The present TRISS software needs the UTM coordinate data to estimate the scoring angle with respect to the aircraft that deployed the bomb (i.e. ingress angle).

It should be noted that the TRISS software could be modified to only use the distance between C1 and C2 and the cameras internal angles to estimate the target position and the scoring error. This could be accomplished by rotating the scoring axis in such a manner that C1 and the target lie on a y-prime axis. The x-prime axis will be perpendicular to the y-prime axis. The scoring error will then be computed with respect to this axis. The scoring error computed in this new artificial reference frame should be identical to a scoring error computed on a UTM axis. The xy-prime axis will then have to be rotated to match the UTM axis to compute the scoring angle. The beauty of this approach is that the error due to erroneous UTM positions for C1 and C2 will be on the scoring angle but not on the actual scoring distance.

4.2 Case 2 Error Model

In Case 2 the mathematical model assumes that the user has measured the UTM positions of C1, C2 and the Target. The model assumes a CEP measurement error of approximately 15 feet. The program user has the capability to change this parameter at will.

4.3 Case 3 Error Model

In Case 3 the mathematical model assumes that the angles between C1 and C2 and the target were measured with an accuracy of approximately 1 degree (i.e. 1 sigma). The model also assumes that the distance between the C1 and C2 is also known with an accuracy of 5 feet. The user has the capability to change these two parameters. The model also assumes that an imprecise GPS device was also used to measure the location of C1 and C2. The model also assumes that a triangulation algorithm is used to estimate the target position.

4.4 Case 4 Error Model

In Case 4 the mathematical model assumes that the angles between C1 and C2 and the target were measured with an accuracy of approximately 1/5th degree. The model also assumes that the distance between C1 and C2 is known with an accuracy of 1 foot.

5. ANALYSIS RESULTS

5.1 Case 1 Analysis

Figure 1, subplots 1 and 2 depict the expected TRISS errors under different geometries on two different scales. The x-axis is the number of degrees between C1 and C2. The y-axis represents the standard deviation of the measurement error. The plot shows the standard deviation errors when C2 rotates around the target. Notice that the errors increase exponentially when C2 is exactly opposite to C1 (0 degrees) and when C2 and C1 are on the same spot (180 degrees). Figure 1 shows that the best accuracy is given when the angular separation between C1 and C2 is within 80-130 degree range. These analysis results make sense since good geometry is needed to estimate the target position. If C1, C2, and the target are in the same plane the geometry is poor and the errors are considerably amplified.

5.2 Case 2 Analysis

Figure 2, subplots 1 and 2 show the same data as Figure 1 but for test case #2. As described before, in this case the measured UTM positions of C1, C2 and the Target were used to estimate the internal angles and the target position. Notice that in this case the TRISS errors only increased exponentially when C1 and C2 shared the same position. This can be explained by the fact that, for Case 2, slightly inaccurate UTM coordinates were used to estimate the internal angles. The ANGLE_C used to estimate the target position as indicated in Equation 1 was never exactly 180 degrees as it was for Case 1

where exact measurements for the internal angles were used to estimate the target position. Notice that large errors may be generated when ANGLE_C is close to 0 or 180 degrees.

C1_T = C1_C2 * sin(d2r*ANGLE_B) / sin(d2r*ANGLE_C);   (Equation 1)

5.3 Case 3 Analysis

For this case, it was assumed that the internal angles were measured with an accuracy of 1 degree (standard deviation = 1 degree) and the distance between C1 and C2 had en error of 5 feet. Notice how sensitive TRISS is to angular inaccuracies. Figures 3 and 4 show that a 1-degree of standard deviation error will generate a 60 feet scoring error under the best geometry. Figure 3 also assumes a 5 feet error in the measured distance between Camera 1 and Camera 2. Case 1 showed a 21 feet scoring error when perfect measurements were assumed for the internal angles. That is, a 15 feet error in the measurement of the C1 and C2 positions led to a 21 feet error in the estimation of the target position under ideal geometries. This gives you a GDOP (geometry dilution of precision) factor of 1.3. It should be noted that the GDOP increases by a factor of 2.5 when a 1-degree error is introduced into the equation

5.4 Case 4 Analysis

Figure 5 shows the TRISS errors for a case where the distance between C1 and C2 is precisely known within 1 foot and the internal angles are known with an accuracy of $1/5^{th}$ of a degree.

6. SUMMARY OF RESULTS AND RECOMENDATIONS

6.1 The analysis shows that the best geometry is obtained when the cameras C1 and C2 are placed 80-130 degrees apart.

6.2 As described in Case 3, TRISS is extremely sensitive to angular inaccuracies.

6.3 Under the best geometry an estimated 21 feet of error is expected if the location of the cameras C1 and C2 is obtained using a standard hand-held GPS.

6.4 It is strongly recommended that the positions of C1 and C2 be properly surveyed using specialized GPS surveying equipment. Under ideal geometry every foot of error will generate a TRISS scoring error of approximately 1.4 feet.

6.5 It is also recommended that internal angles be also accurately measured to $1/5^{th}$ of a degree. The mathematical model shows acceptable scoring errors (i.e. < 10 feet) when angular accuracies of $1/5^{th}$ of a degree are assumed.

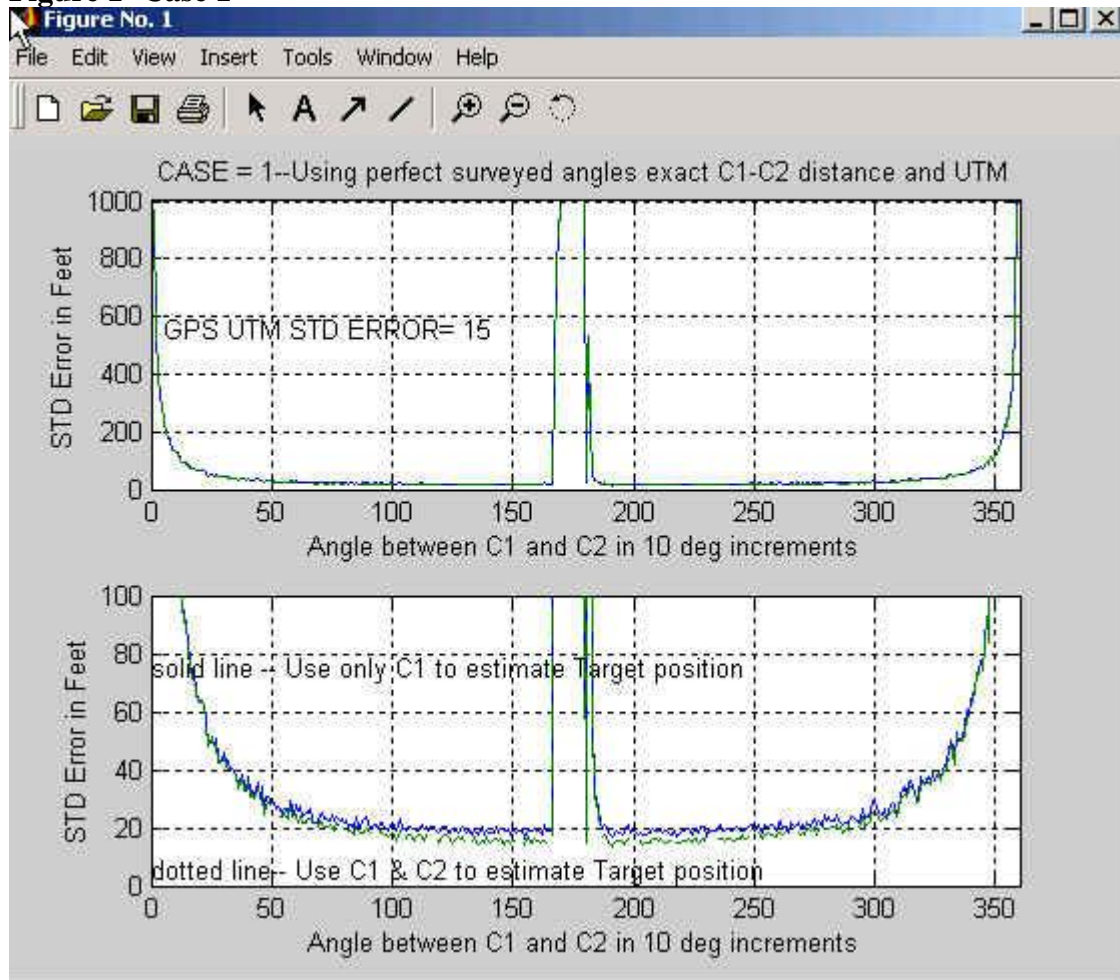# APPENDIX A -- TRISS MATLAB PLOTS
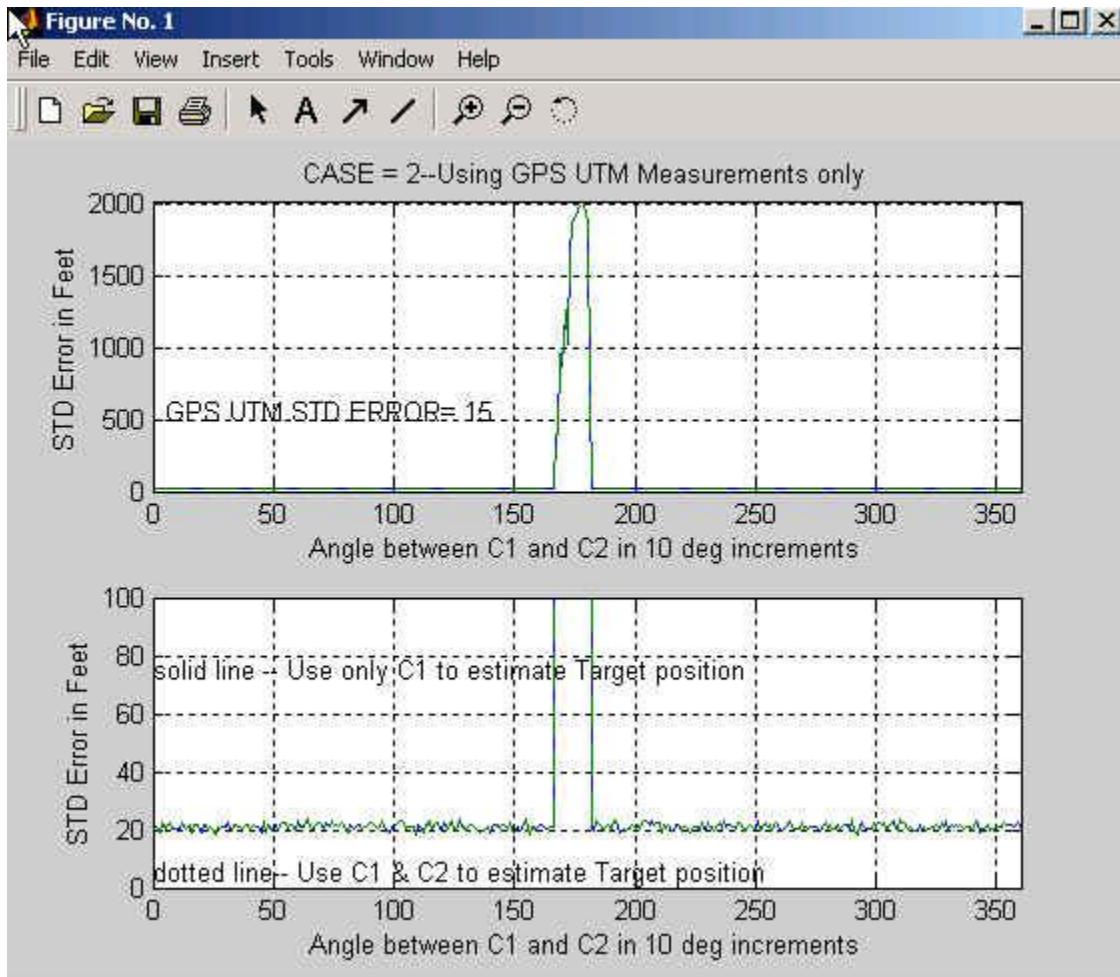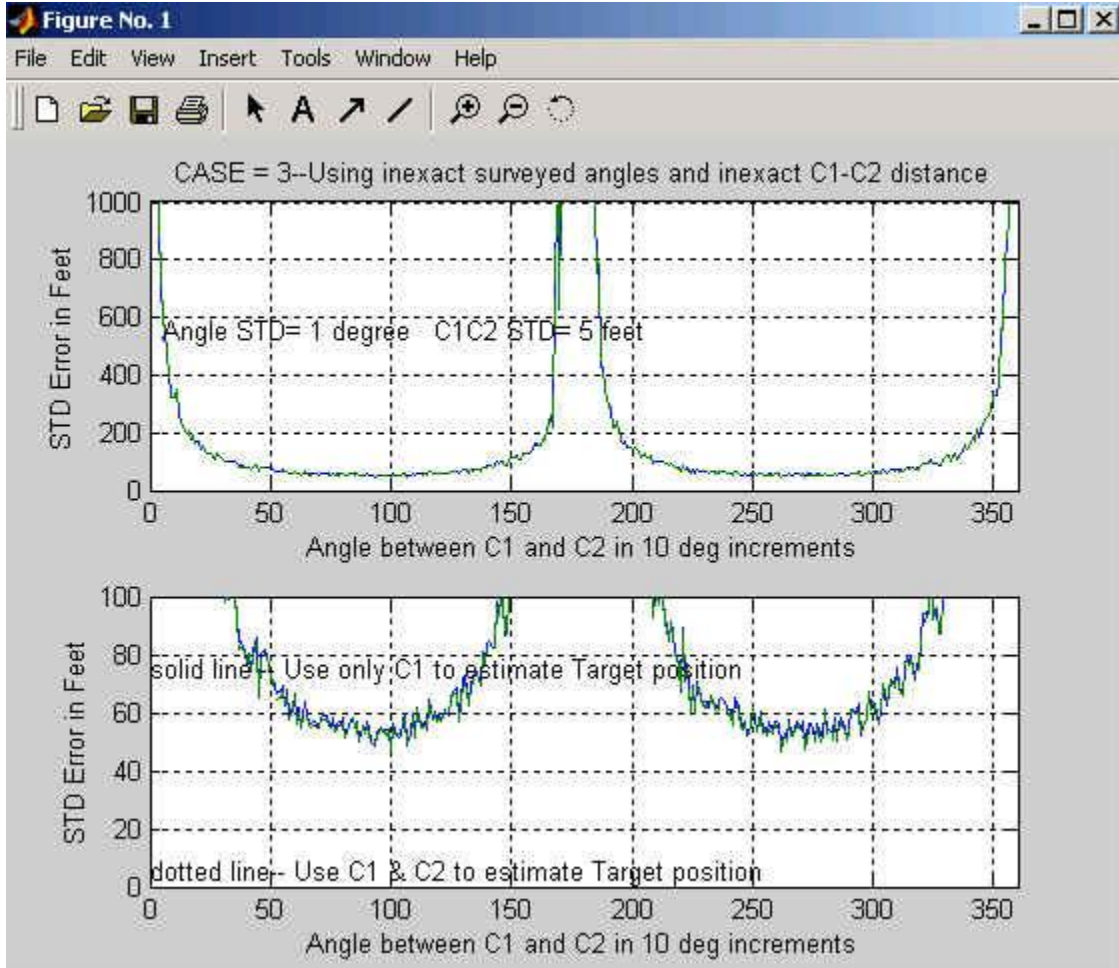
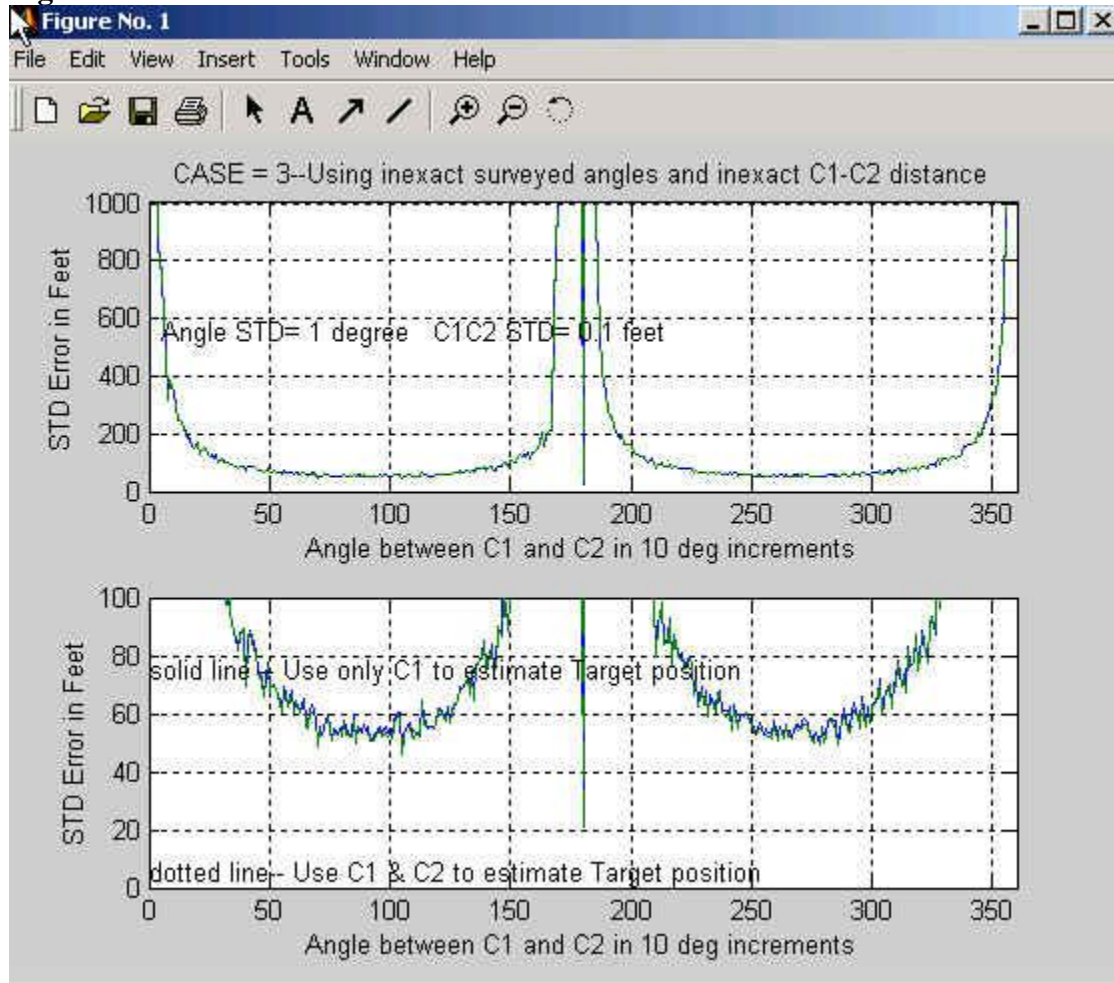**Figure 1- Case 1**

Figure 2 – Case 2

Figure 3 – Case 3

Figure 4 Case 4

Figure 5 Case 5
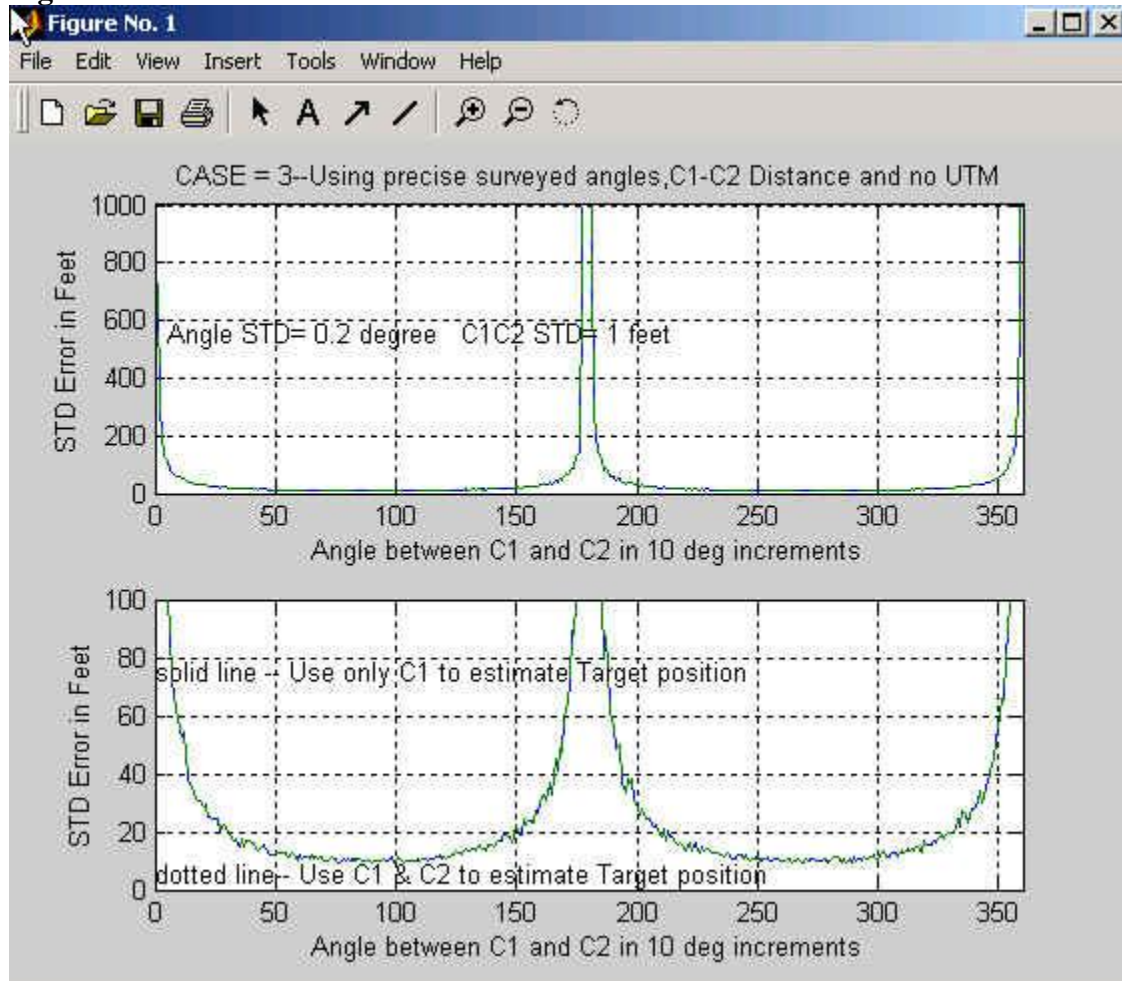
APPENDIX B -- MATLAB PROGRAM

```
clear all
%*************************************************************
%  TRISS ERROR SIMULATION MODEL(S)
%  By Luis E. Alvarado -------------- 4/27/02
%
%  Case 1 -- Exact disctance between cameras 1 and 2 is known and the exact
%           internal angles from C1 to C2 are known. Not known are the
%           exact UTM coordinates for C1,C2 and Target.
%  Case 2 -- All angles and C1 to C2 distance are computed exclusively using UTM
%  Case 3 -- Same as Case 1 except that a STD error of "N" degrees is assumed in the
%           measurement of
%           of the internal angles and a STD of "X" feet is assumed in the measurement of
%           the C1_C2 distance.
%*************************************************************
r2d=180/pi;
d2r=pi/180;
theta = 0;
randn('state',sum(100*clock)) %resets random to a different state each time.
%*************************************************************
% USER INPUTS
%*************************************************************
position = 2;      %=2 T is in center. C1 2000 feet to the left of T.C2 2000 ft to the right of T
                   %=1 C1 is in the center, T 2000 ft to the right of C1. C2 2000 ft to the rt of T
gpserr =  15.0;    %15 feet of error
angleErr = 1.0;    %1 degree
c1c2Err  = 5.0;    %5 feet
CaseNum = input(' Please enter Error Analysis Case number = ');

m=1:360;                % number of Camera 2 positions
n=100;                  % Number of Monte Carlo runs per C2 position
for j=m
theta = theta + 1;        %rotate C2 position. Leave C1 at (0,2000) and target in center
if position==1
   PX1_utm =0;            % UTM x position of Camera 1
   PY1_utm =0;            % UTM y position of Camera 1
   PXt_utm = 2000;        % UTM x position of Target
   PYt_utm = 0;           % UTM y poistion of Target
elseif position==2
   PX1_utm =-2000;        % UTM x position of Camera 1
   PY1_utm =0;            % UTM y position of Camera 1
   PXt_utm =0;            % UTM x position of Target
   PYt_utm =0;            % UTM y poistion of Target
end

C1_R = sqrt((PXt_utm - PX1_utm)^2 + (PYt_utm - PY1_utm)^2);
PX2_utm = C1_R * cos(theta * d2r) + PXt_utm;
PY2_utm = C1_R * sin(theta * d2r) + PYt_utm;

%***********************************************************************
%   Start Monte Carlo Simulation
%   Assume GPS UTM Error of 15 feet in x-y plane
%***********************************************************************
for i=1:n

   X1_utm = PX1_utm +  randn * gpserr;
   Y1_utm = PY1_utm +  randn * gpserr;
```

```matlab
   X2_utm = PX2_utm +  randn * gpserr;
   Y2_utm = PY2_utm +  randn * gpserr;
   Xt_utm = PXt_utm +  randn * gpserr;
   Yt_utm = PYt_utm +  randn * gpserr;


%-------------------------------------------------------------------------------
%  If Case 1 -- Compute external angles using precisely suveyed internal angles and surevey
%  distance between C1 and C2 and GPS measured C1 and C2 UTM coordinate data;
%-------------------------------------------------------------------------------
 if CaseNum == 1
%***************************************************************
% Assume the Internal angles and the distance between C1 and C2 are
% precisely known. Zero measurement error is assumed.
%***************************************************************
   dx21 = PX2_utm - PX1_utm;
   dy21 = PY2_utm - PY1_utm;
%------------------------------------------------------
% Compute exact distance between Camera 1 and Camera 2
   C1_C2 = sqrt( dx21.^2 + dy21.^2 );
% Use UTM to compute Angle between C1 and C2
   dx21 = X2_utm - X1_utm;
   dy21 = Y2_utm - Y1_utm;

   BETA_C = atan2(dx21, dy21)*r2d; %Angle from C1 to C2
   if BETA_C < 0
      BETA_C = BETA_C + 360;
   end
   %------------------------------------------------------
   dxt1 = PXt_utm - PX1_utm;
   dyt1 = PYt_utm - PY1_utm;
   ALPHA_T = atan2(dxt1, dyt1)*r2d; %Angle from C1 to target
   if ALPHA_T < 0
      ALPHA_T = ALPHA_T + 360;
   end
   %------------------------------------------------------
   dxt2 = PXt_utm - PX2_utm;
   dyt2 = PYt_utm - PY2_utm;

   GAMA_T = atan2(dxt2, dyt2)*r2d; %Angle form C2 to target
   if GAMA_T < 0
      GAMA_T = GAMA_T + 360;
   end
   %------------------------------------------------------
   aft1 = ALPHA_T +180;
   if aft1 >= 360
      aft1 = aft1 - 360;
   end
   aft2 = GAMA_T + 180;
   if aft2 >= 360
      aft2 = aft2 - 360;
   end
   ANGLE_C = abs(aft1 - aft2);
   if ANGLE_C > 180
      ANGLE_C = 360 - ANGLE_C;
   end
```

```matlab
    ANGLE_A = abs(BETA_C - ALPHA_T);
    if ANGLE_A  < 0
       ANGLE_A = 360 + ANGLE_A;
    end
    ANGLE_B = 180 - ANGLE_C - ANGLE_A;
%***********************************************************************
elseif CaseNum==2
%-------------------------------------------------------------------------------
%  If Case 2 -- Compute external angles and internal angles and C1_C2 distance
%  exclusively using C1, C2 and Target GPS UTM coordinate data.
%-------------------------------------------------------------------------------
    dxt1 = Xt_utm - X1_utm;
    dyt1 = Yt_utm - Y1_utm;

    ALPHA_T = atan2(dxt1, dyt1)*r2d; %Angle from C1 to target

    if ALPHA_T < 0
       ALPHA_T = ALPHA_T + 360;
    end
%-------------------------------------------------------
    dxt2 = Xt_utm - X2_utm;
    dyt2 = Yt_utm - Y2_utm;

    GAMA_T = atan2(dxt2, dyt2)*r2d; %Angle form C2 to target
    if GAMA_T < 0
       GAMA_T = GAMA_T + 360;
    end


%-------------------------------------------------------
    dx21 = X2_utm - X1_utm;
    dy21 = Y2_utm - Y1_utm;
    BETA_C = atan2(dx21, dy21)*r2d; %Angle from C1 to C2
    if BETA_C < 0
       BETA_C = BETA_C + 360;
    end
%-------------------------------------------------------
    aft1 = ALPHA_T +180;
    if aft1 >= 360
       aft1 = aft1 - 360;
    end
    aft2 = GAMA_T + 180;
    if aft2 >= 360
       aft2 = aft2 - 360;
    end
    ANGLE_C = abs(aft1 - aft2);
    if ANGLE_C > 180
       ANGLE_C = 360 - ANGLE_C;
    end
    ANGLE_A = abs(BETA_C - ALPHA_T);
    if ANGLE_A  < 0
       ANGLE_A = 360 + ANGLE_A;
    end
    ANGLE_B = 180 - ANGLE_C - ANGLE_A;
%-------------------------------------------------------
    C1_C2 = sqrt( dx21.^2 + dy21.^2 );
elseif CaseNum == 3
```

```
%***********************************************************************
% If Case 3 -- Same as Case 1 except that a STD error of "N" degrees is assumed in the
measurement of
% of the internal angles and a STD of "X" feet is assumed in the measurement of the C1_C2
distance.
%***********************************************************************
   dx21 = PX2_utm - PX1_utm;
   dy21 = PY2_utm - PY1_utm;
%----------------------------------------------------
% Compute the distance between Camera 1 and Camera 2
   C1_C2 = sqrt( dx21.^2 + dy21.^2 ) + randn*c1c2Err;
% Use UTM to compute Angle between C1 and C2
   dx21 = X2_utm - X1_utm;
   dy21 = Y2_utm - Y1_utm;

   BETA_C = atan2(dx21, dy21)*r2d; %Angle from C1 to C2
   if BETA_C < 0
      BETA_C = BETA_C + 360;
   end
   %----------------------------------------------------
   dxt1 = PXt_utm - PX1_utm;
   dyt1 = PYt_utm - PY1_utm;
   ALPHA_T = atan2(dxt1, dyt1)*r2d + randn*angleErr; %Angle from C1 to target
   if ALPHA_T < 0
      ALPHA_T = ALPHA_T + 360;
   end
   %----------------------------------------------------
   dxt2 = PXt_utm - PX2_utm;
   dyt2 = PYt_utm - PY2_utm;

   GAMA_T = atan2(dxt2, dyt2)*r2d + randn*angleErr; %Angle form C2 to target
   if GAMA_T < 0
      GAMA_T = GAMA_T + 360;
   end
   %----------------------------------------------------
      aft1 = ALPHA_T +180;
   if aft1 >= 360
      aft1 = aft1 - 360;
   end
   aft2 = GAMA_T + 180;
   if aft2 >= 360
      aft2 = aft2 - 360;
   end
   ANGLE_C = abs(aft1 - aft2);
   if ANGLE_C > 180
      ANGLE_C = 360 - ANGLE_C;
   end
   ANGLE_A = abs(BETA_C - ALPHA_T);
   if ANGLE_A  < 0
      ANGLE_A = 360 + ANGLE_A;
   end
   ANGLE_B = 180 - ANGLE_C - ANGLE_A;
else
   [' INVALID CASE SELECTION -- IGNORE RESULTS ']
end
```

```matlab
%Compute Camera 1 to target distance  C1_T
   if sin(d2r*ANGLE_C) == 0
      ANGLE_C = ANGLE_C + 0.00001;
   end
   C1_T = C1_C2 * sin(d2r*ANGLE_B)/sin(d2r*ANGLE_C);
%Compute Camera 2 to target distance   C2_T
   C2_T = C1_C2 * sin(d2r*ANGLE_A)/sin(d2r*ANGLE_C);
% Estimate target position using Camera 1
   Xt1_utm = X1_utm + C1_T * sin(ALPHA_T*d2r);
   Yt1_utm = Y1_utm + C1_T * cos(ALPHA_T*d2r);
% Estimate target position using Camera 2
   Xt2_utm = X2_utm + C2_T * sin(GAMA_T*d2r);
   Yt2_utm = Y2_utm + C2_T * cos(GAMA_T*d2r);
% Average target position
   Xtavg = (Xt1_utm + Xt2_utm)/2;
   Ytavg = (Yt1_utm + Yt2_utm)/2;
%*******************************************************************
%  Save variables
%*******************************************************************
   Xt_err(i) = PXt_utm - Xt1_utm;
   Yt_err(i) = PYt_utm - Yt1_utm;

   Xt_Aerr(i) = PXt_utm - Xtavg;
   Yt_Aerr(i) = PYt_utm - Ytavg;
end
%------------------------------------------------------
%   SAVE VARIABLE
%------------------------------------------------------
if CaseNum == 1
  ALPHA_S(j)= ALPHA_T;
  GAMA_S(j) = GAMA_T;
%--------------------------------
  Xt1_S1(j) = Xt1_utm;
  Yt1_S1(j) = Yt1_utm;
  Xt2_S1(j) = Xt2_utm;
  Yt2_S1(j) = Yt2_utm;
else
  Xt1_S2(j) = Xt1_utm;
  Yt1_S2(j) = Yt1_utm;
  Xt2_S2(j) = Xt2_utm;
  Yt2_S2(j) = Yt2_utm;
end
  deg(j) = theta;
  TM_err(j) = sqrt( std(Xt_err)^2 + std(Yt_err)^2 );
  TM_Aerr(j) = sqrt( std(Xt_Aerr)^2 + std(Yt_Aerr)^2 );

end
%*************** END OF MONTECARLO SIMULATION LOOP ********************
%*********************************************************************
% PLOT RESULTS
%*********************************************************************
if CaseNum==1
  tt2=['--Using perfectly surveyed angles and exact C1-C2 distance'];
  max1=1000;
elseif CaseNum==2
  tt2=['--Using GPS UTM Measurements only'];
```

```
    max1=2000;
else
    tt2=['--Using inexact surveyed angles and inexact C1-C2 distance'];
    max1=1000;
end
tt3=['Angle between C1 and C2 in 10 deg increments'];
tt=['CASE = ',num2str(CaseNum),tt2];
tp1 = ['GPS UTM STD ERROR= ' num2str(gpserr)];
tp2 = ['Angle STD= ' num2str(angleErr) ' degree' '   C1C2 STD= ' num2str(c1c2Err) ' feet'];

    subplot(211)
    plot(deg,TM_err,deg,TM_Aerr,'--'),title(tt),xlabel(tt3),grid;
    if CaseNum < 3
        text(5,550,tp1);
    else
        text(5,550,tp2);
    end

    ylabel('STD Error in Feet')
    axis([0 360 0 max1]);
%---------------------------
    subplot(212)
    plot(deg,TM_err,deg,TM_Aerr,'--'),xlabel(tt3),grid;
    text(0,75,'solid line -- Use only C1 to estimate Target position');
    text(0,5,'dotted line-- Use C1 & C2 to estimate Target position');
    ylabel('STD Error in Feet')
    axis([0 360 0 100]);
    subplot;
```

TRISS ERROR ANALYSIS

Part II – More Analysis Cases

Luis E. Alvarado

OUTLINE

1. INTRODUCTION

1.1 Scope

This document summarizes the analysis conducted to evaluate the susceptibility of the Tracking Infrared Scoring System (TRISS) to site measurement errors and camera calibration errors for different TRISS configurations.

1.2 Background

TRISS is a ground based scoring system that uses two cameras with infrared lenses to determine the distance between a pre-selected target and the position where the deployed warhead hits the ground. TRISS also determines the scoring angle with respect to the aircraft that deployed the bomb (i.e. ingress angle).

2. CONCLUSION

The analysis shows that the accuracy of the scoring error is totally dependent in the accuracy of the internal angles between the cameras and the target used for scoring and the accuracy of the position of Cameras 1 and 2. The analysis also shows that the absolute position of C1 and C2 is not necessarily needed as long as the distance between C1 and C2 is accurately known and TRISS "knows" where true North is. The analysis also indicates that TRISS is very sensitive to Camera calibration errors.

3. TRISS MATHEMATICAL MODEL

The analysis was conducted using a Monte-Carlo simulation model of TRISS. This model was developed using the MATLAB program. The MATLAB program is a 4th generation language that provides the user with the necessary tools to easily generate math models of complex physical systems.

The TRISS mathematical model assumes that the target is co-centric to cameras 1 and 2. Camera 1 position is fixed and Camera 2 rotates around the target. The TRISS model computes the estimated target position error every time C2 is rotated one degree around the target. The user has the capability to modify C1 and C2 starting positions as well as the radius between C1 and the target. The user also has the capability to vary camera position and add angular measurement errors. The user also has the capability to simulate camera calibration errors.

4. ANALYSIS RESULTS

4.1 Case 1 Analysis

For this case scenario, the cameras were positioned 90 degrees apart. The target was in the center approximately 1 mile from cameras 1 and 2. It was assumed that both cameras experienced an angular misalignment standard deviation error (SDE) of 0.1degrees with the respect to the target. Figure 1-1, subplots 1 and 2 depict the expected TRISS error estimating target position under different geometries on two different scales. The x-axis is the number of degrees that Camera 2 is rotated (counterclockwise) around the target. The y-axis represents the standard deviation of the measurement error. The plot shows the standard deviation errors when C2 is rotated around the target and C1 remains stationary. Notice that the errors increased exponentially when C2 was exactly opposite to C1 (0 degrees) and when C2 and C1 were on the same spot (180 degrees). Figure 1-1 shows that the best accuracy is given when the angular separation between C1 and C2 is within 80-130 degree range. These analysis results make sense since good geometry is needed to estimate the target position. If C1, C2, and the target are in the same plane the geometry is poor and the errors are considerably amplified. Figure 1-2 shows the internal angles A, B and C of triangle formed by C1, C2 and the target. Figure 1-3 shows the distances estimated by TRISS between C1 and the Target and between C2 and the target using equations 1 and 2.

> **C1_T = C1_C2 * sin(d2r*ANGLE_B) / sin(d2r*ANGLE_C);   (Equation 1)**

> **C2_T = C1_C2 * sin(d2r*ANGLE_A) / sin(d2r*ANGLE_C);   (Equation 2)**

Notice in Figure 1-3 that the computation of the estimated distances became "unstable" when ANGLE_C approaches 0 degrees and when ANGLE_A or ANGLE_B approaches 0 degrees at the same time ANGLE_C approaches 0 degrees (system singularities). Remember the ratios k/0, and 0/0 are numerically unstable. If TRISS becomes a dynamic system it is recommended to eliminate these singularity points.

Figure 1-4 shows the TRISS scoring error when the "bomb" is rotated 100 feet around the target. A 0.1degree standard deviation calibration error was assumed for both cameras and as before, a 0.1 degree angular misalignment SDE was assumed between both cameras and the target. Notice the scoring error is pretty much constant (~ 20 feet) around the target. Figure 1-5 shows the internal angles A, B and C of the triangle formed by C1, C2 and the bomb. Notice that the angles barely change when the bomb is rotated around the target. Notice also that the internal angles never approached 0 degrees. This explain why the scoring error is pretty constant around the target as depicted in Figure 1-4

Figure 1-6 illustrates an X-Y plot indicating the position of C1 and C2 with respect to the target. Notice the cameras were positioned exactly 90 degrees apart.

Figure 1-7 shows the trajectory of the bomb rotating around the target (100 feet radius circle) and the multiple positions predicted by the TRISS Monte Carlo simulation model.

4.2 Case 2 Analysis

This simulation case is identical to Case 1 with the exception that the Cameras were positioned 40 degrees apart with respect to the target. This was done to show the importance of placing the cameras as close as possible 90 degrees with respect to the target. Figure 2-1 shows the TRISS scoring error when the "bomb" is rotated 100 feet around the target. As before a 0.1degree standard deviation calibration error was assumed for both cameras and a 0.1 degree angular misalignment SDE was assumed between both cameras and the target. Notice the scoring error is no longer constant and at some scoring angles, the scoring error was close to 40 feet.

Figure 2-2 illustrates an X-Y plot indicating the position of C1 and C2 with respect to the target. Notice the cameras were positioned exactly 40 degrees apart.

Figure 2-3 shows the trajectory of the bomb rotating counterclockwise around the target (100 feet radius circle) and the multiple positions predicted by the TRISS Monte Carlo simulation model. Notice that the scoring inaccuracies were almost doubled by repositioning the cameras closer together (From 90 degrees to 40 degrees).

4.3 Case 3 Analysis

For this case scenario, the angular misalignment errors between C1 and the target and between C2 and the target were varied from 0 degrees to 0.5 degrees. The intention of this simulation test was to show how sensitive TRISS is to angular misalignment errors even under perfect geometry situations. C1 and C2 were placed 90 degrees apart with respect to the target. The target was placed 5200 feet from C1 and C2. Figure 3-1 depicts a plot showing TRISS scoring errors in the y-axis versus angular misalignments of C1 and C2 in the x-axis. Notice the error increases from 0 feet to more than 70 feet for 0.5 degrees of angular misalignment. For this experiment zero calibration errors were assumed for C1 and C2.

4.4 Case 4 Analysis

For this case scenario, the angular misalignment errors between C1 and the target and between C2 and the target were assumed to be zero degrees. However, the camera calibration errors were varied from 0 to 1 degree. The intention of this simulation test was to show how sensitive TRISS is to camera calibration errors even under perfect geometry situations. C1 and C2 were placed 90 degrees apart with respect to the target. The target was placed 5200 feet from C1 and C2. Figure 4-1 depicts a plot showing TRISS scoring errors versus camera calibration errors. Notice the error increased from 0 feet to more than 90 feet for a 1 degree standard deviation camera calibration error.

5. SUMMARY OF RESULTS AND RECOMENDATIONS

5.1 The analysis of Cases 1 and 2 show that the best geometry is obtained when the cameras C1 and C2 are placed 80-130 degrees apart.

5.2 Case 3 clearly shows that TRISS is extremely sensitive to angular inaccuracies.

5.3 Case 4 clearly indicates that TRISS is also very sensitive to camera calibration inaccuracies

5.4 Therefore, it is strongly recommended (imperative) that the positions of C1, C2 and the target be properly surveyed using specialized GPS surveying equipment. The total cost of the GPS equipment is approximately $6000 for 3 GPS surveying sites.

5.5 The analysis also shows that the scoring error shall be very constant around the target if the separation angle between the target and C1 and C2 is close to 90 degrees.

APPENDIX A -- TRISS MATLAB PLOTS

Figure 1-1
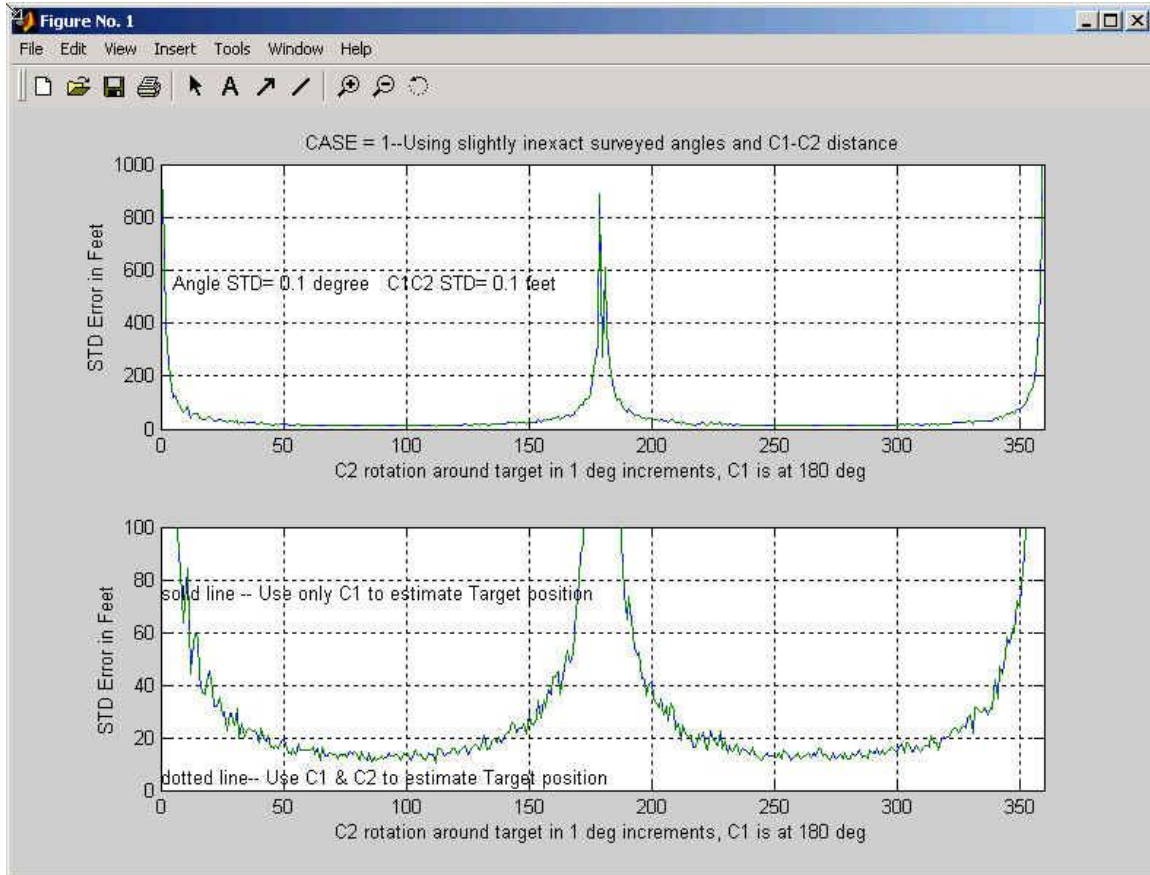
Figure 1-2
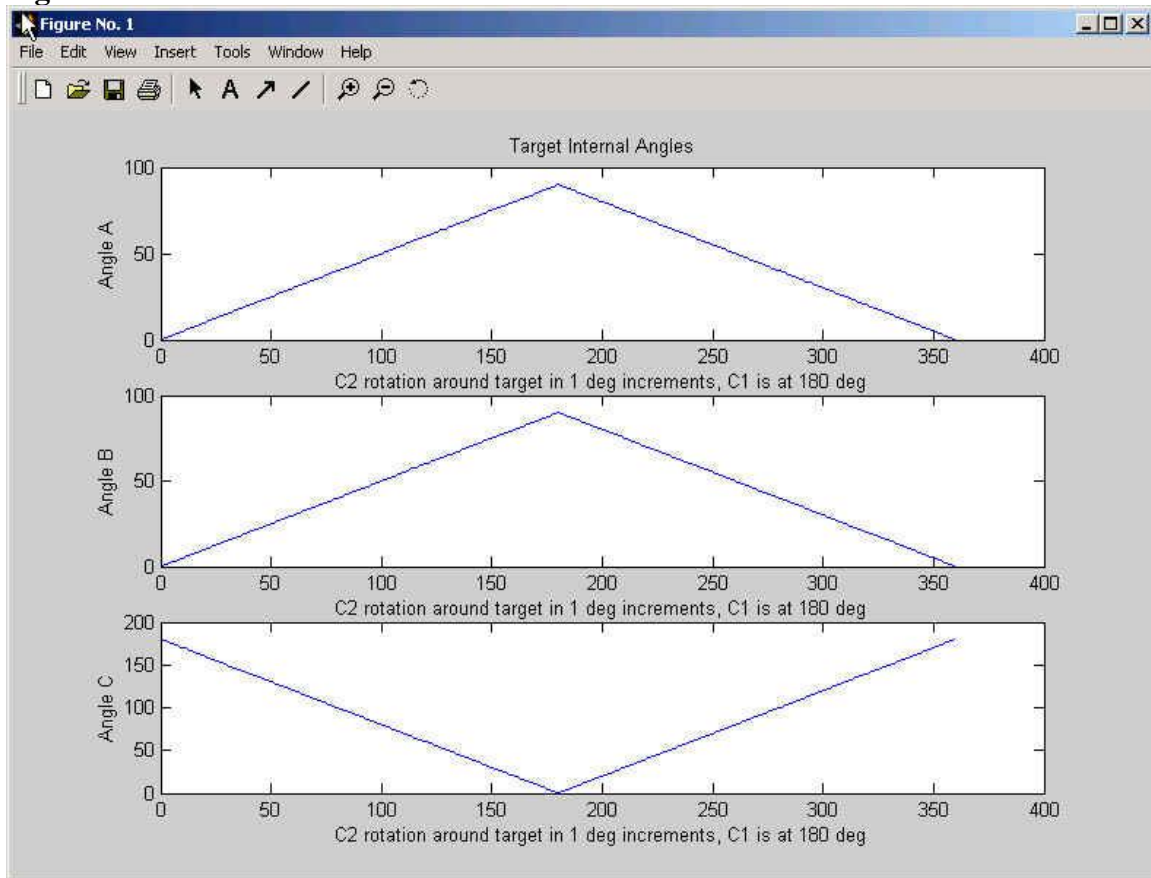
Figure 1-3
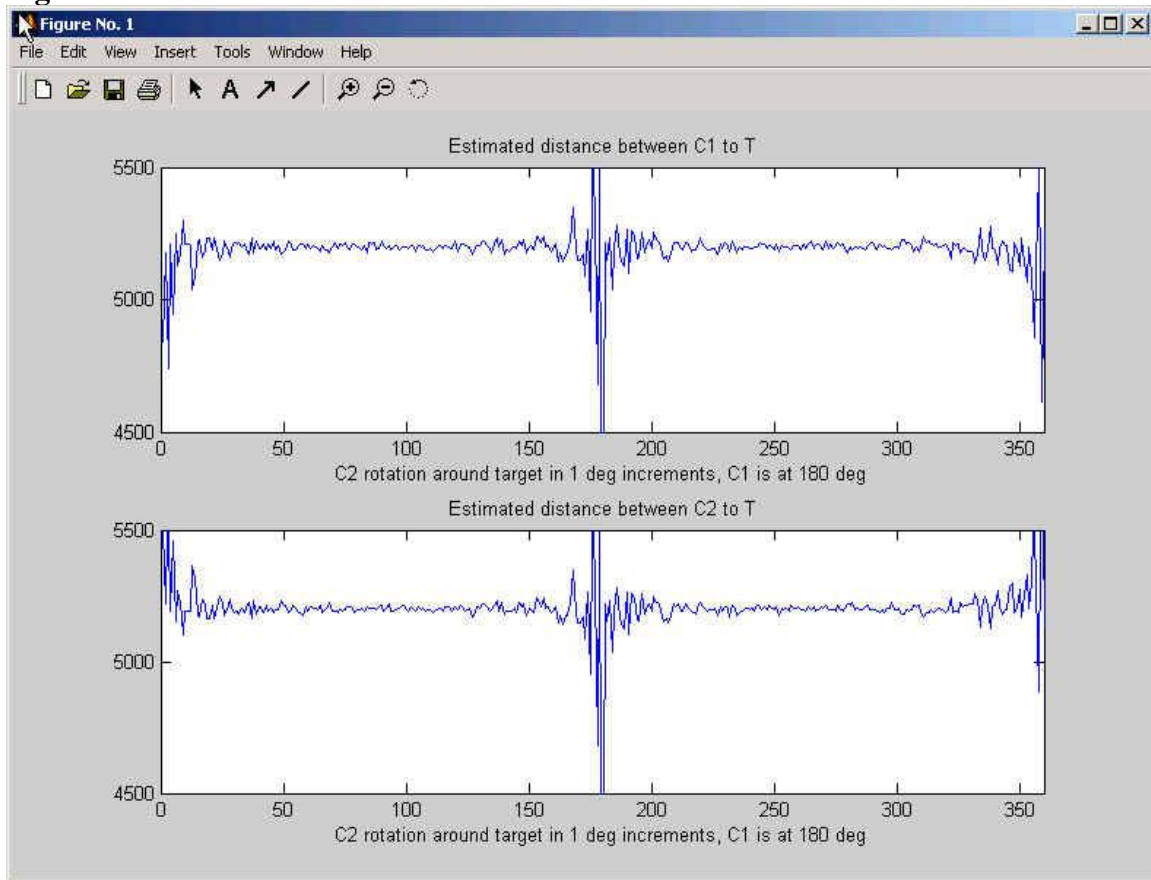
Figure 1-4
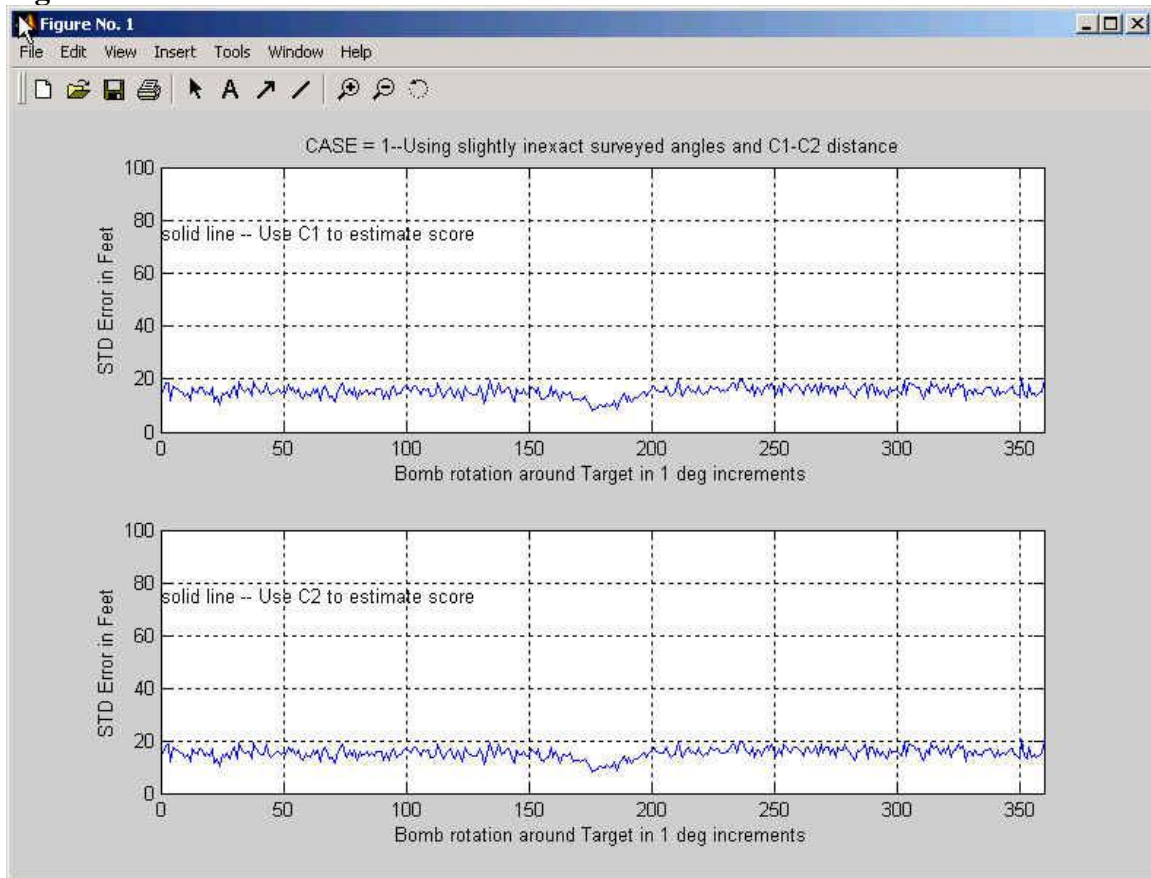
Figure 1-5
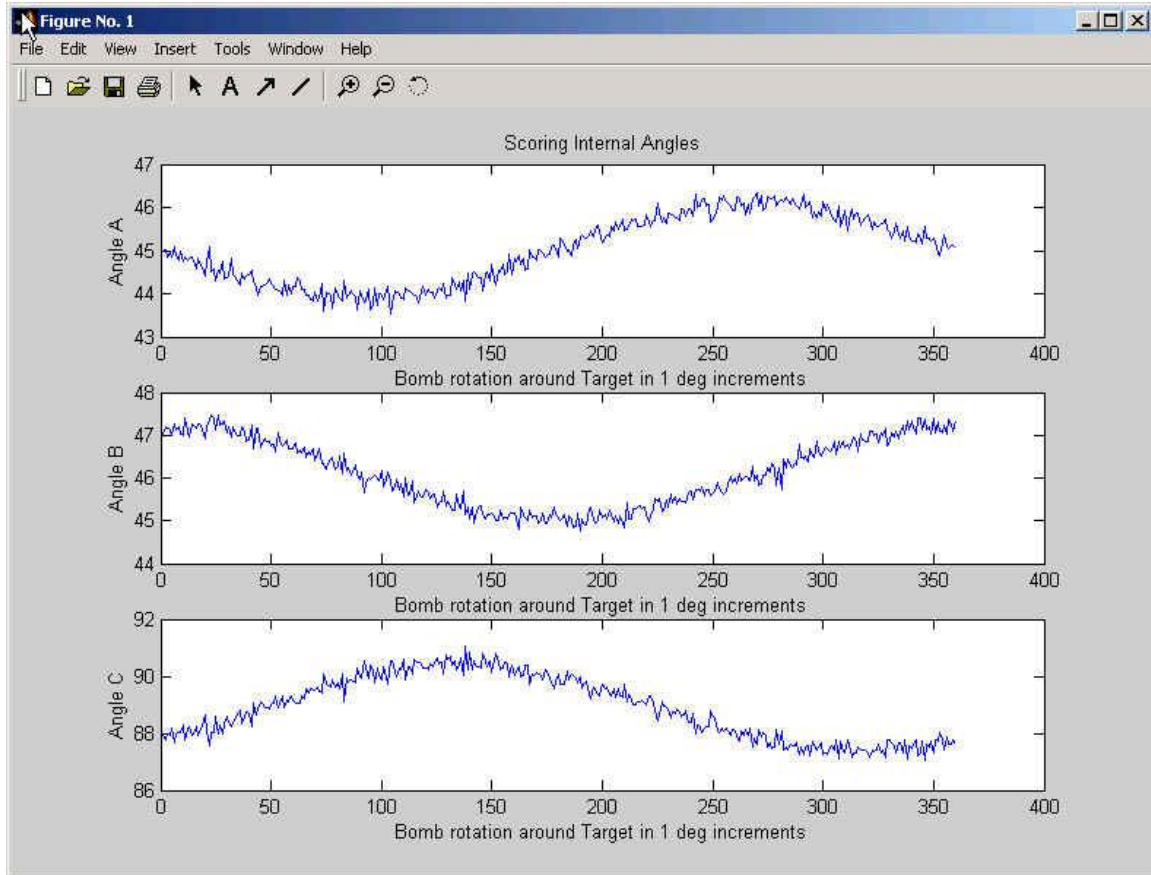
Figure 1-6
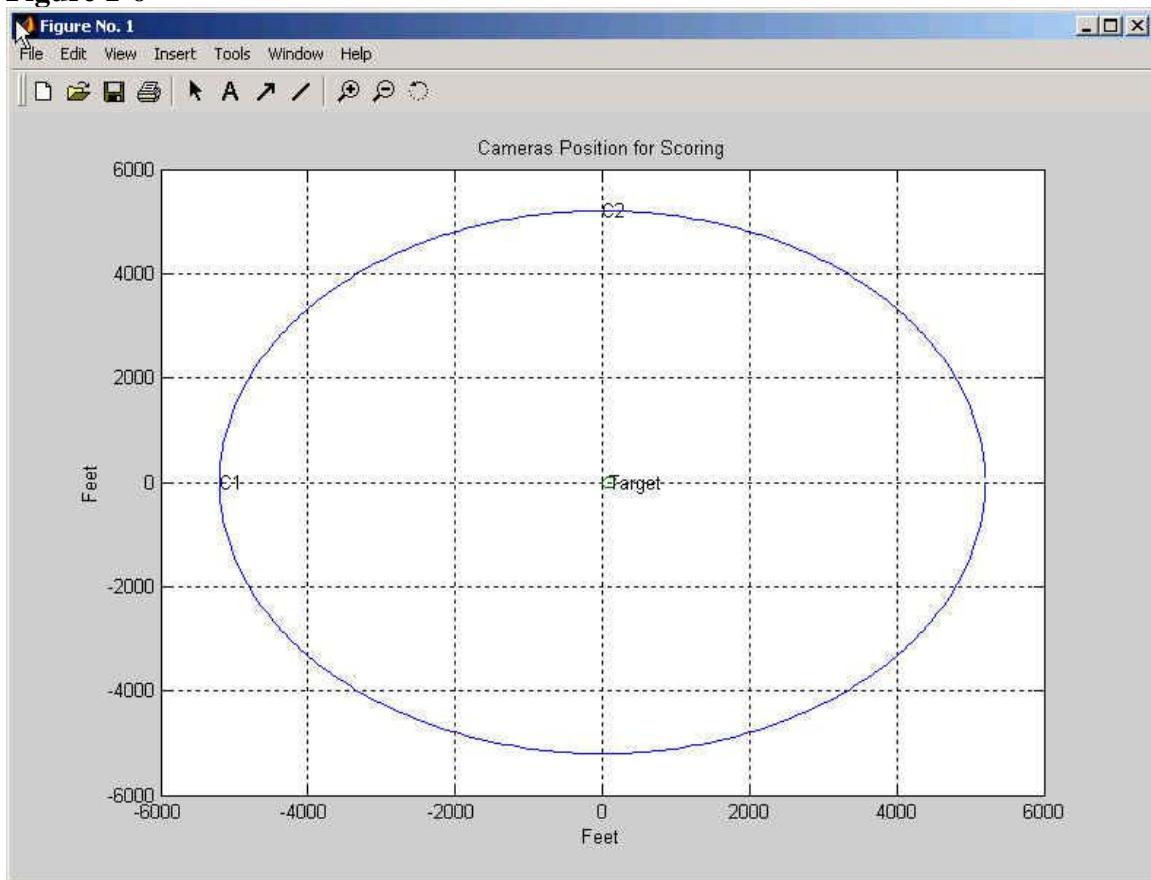
Figure 1-7
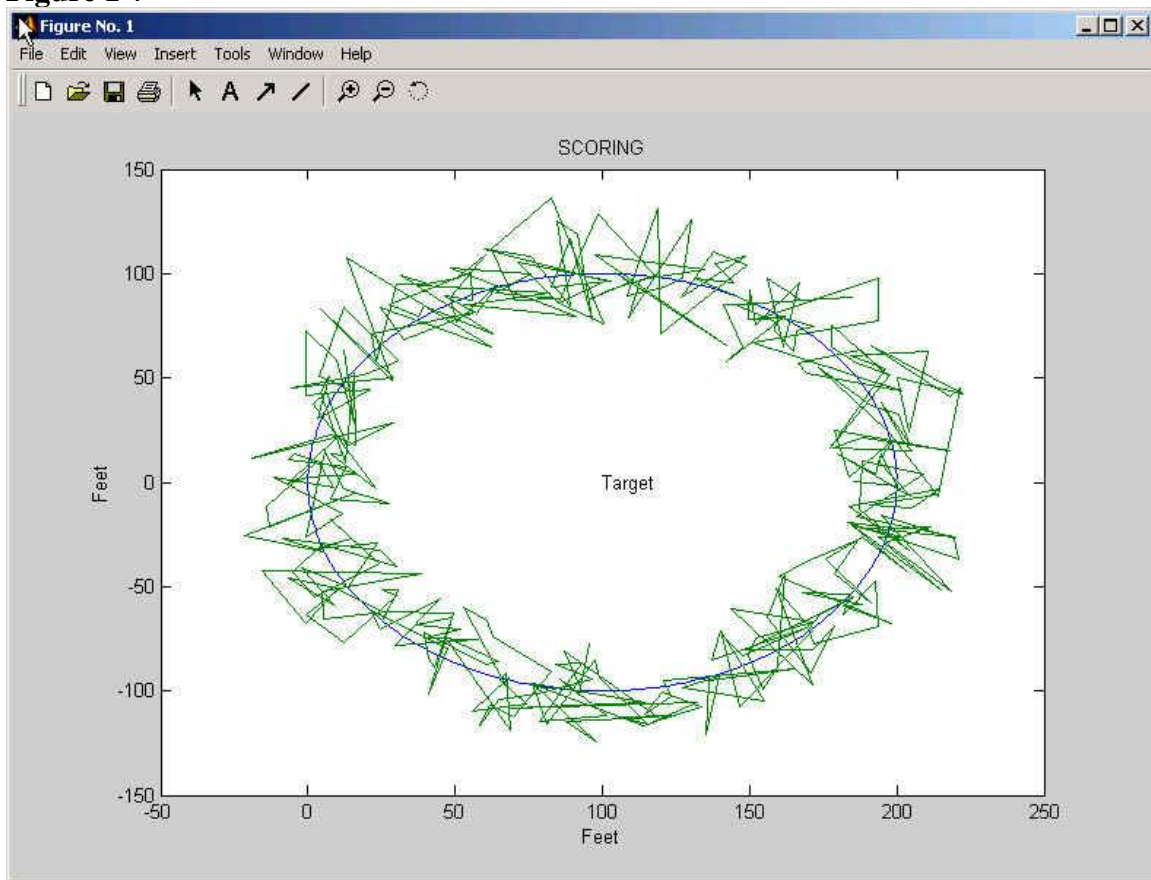
Figure 2-1
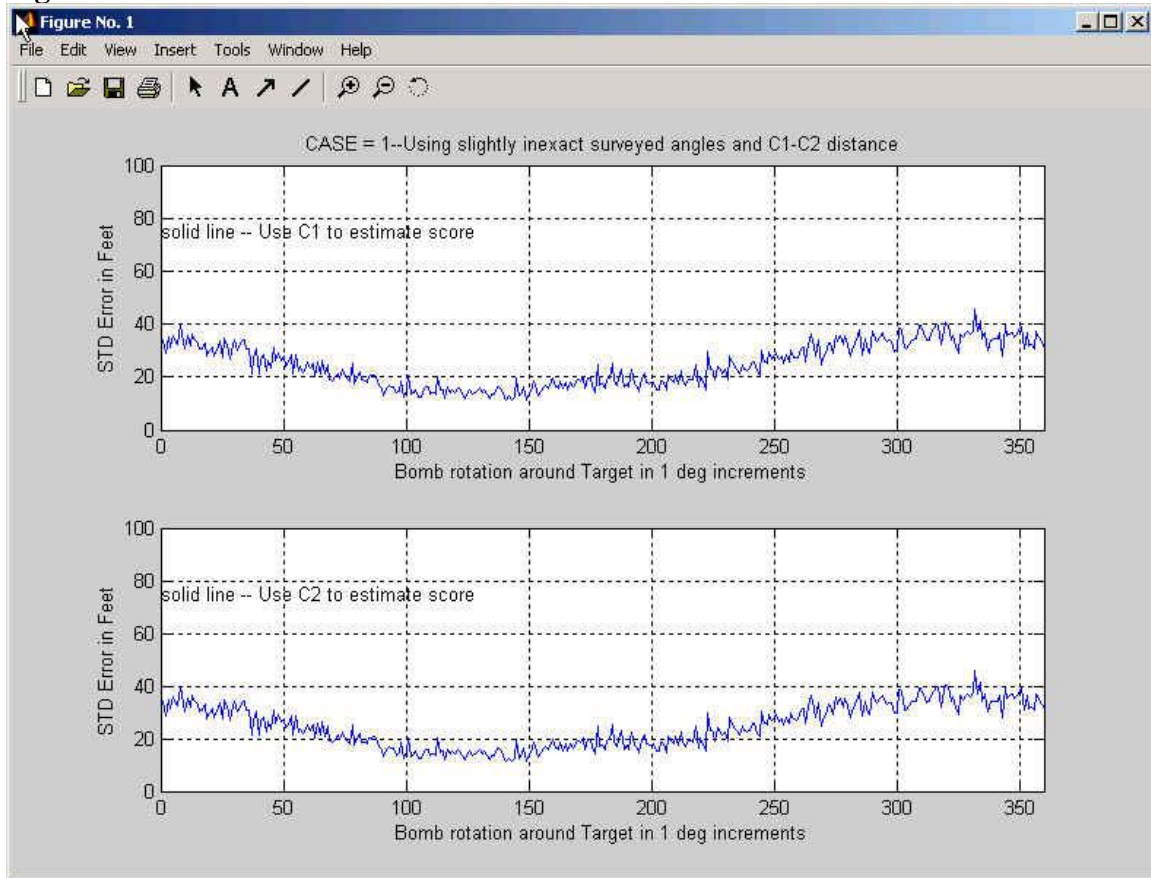
Figure 2-2

Figure 2-3

Figure 3-1

Scoring Error - Angle between target and C1C2 = 90 Degrees

Distance between C1 and Target = 5200 feet

Feet

Angle Errors between target and C1C2 in DEGREES

Figure 4-1