

TRISS Motion Detection Algorithm (TMDA)

By

Luis E. Alvarado

28 April 2003

For

The Proteus Corporation

OUTLINE

1. INTRODUCTION.....	3
1.1 Scope	3
1.2 Background and Motivation.....	3
2. OVERVIEW.....	3
3. DETAILED ALGORITHM DESCRIPTION.....	4
3.1 Motion Detection Algorithm.....	4
3.1.1 Digital Filter	4
3.1.1 Adaptive Threshold Limits	5
3.2 Cluster Detection Algorithm	6
3.3 Event Detection Algorithm.....	9

List of Figures

Figure 1 TMDA Block Diagram.....	4
Figure 2 TMDA Filter Block Diagram.....	4
Figure 3 - Plots of Pixel Intensity filtered data.....	6
Figure 4 Pseudo-Code for the Cluster Detection and Tracking Algorithm.....	7
Figure 5 Cluster Detection Algorithm tracking two real events.....	8
Figure 6 TRISS Images.....	9

1. INTRODUCTION

1.1 Scope

This document describes the Motion Detection Algorithm used by TRISS. Subsection 2 provides a general overview of the algorithm. Subsection 3 provides a detailed description of the algorithm.

1.2 Background and Motivation

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).

The TRISS operator can be overcome by events if he has to manually score consecutive explosive ordinances during a live testing scenario. First, the operator has to visually identify the explosion on the computer displays. Then he has to move the computer mouse and press the mouse's right-hand side push button twice; on the target and where the explosion took place. Furthermore, this operation must be performed for the top display (camera 1) and for the bottom display (camera 2).

The TMDA provides to TRISS the capability to automatically score munitions explosions and laser c-spots in real time. TMDA is capable of detecting and scoring up to 10 nearly simultaneous explosions (1 second apart). It also has the capability to track the motion of up to 10 separate moving objects within the camera(s) Field of View.

2. OVERVIEW

The TMDA is composed of three separate algorithms as described below and illustrated in Figure 1.

Motion Detection Algorithm (MD)

This algorithm processes digital image data from cameras 1 and 2. Uses an adaptive filter to distinguish event data (i.e. explosions) from background clutter.

Cluster Tracking Algorithm (CT)

This algorithm processes the candidate event data detected by the MD algorithm. Detects and tracks the centers of multiple event clusters.

Event Detection Algorithm (ED)

This cross correlation algorithm is used to associate the positions of multiple event clusters detected by the CT algorithm from two or more different cameras for final event detection and scoring. A separate TRISS scoring algorithm compares the cluster detected image position with the image coordinates of the selected stationary target for final scoring.

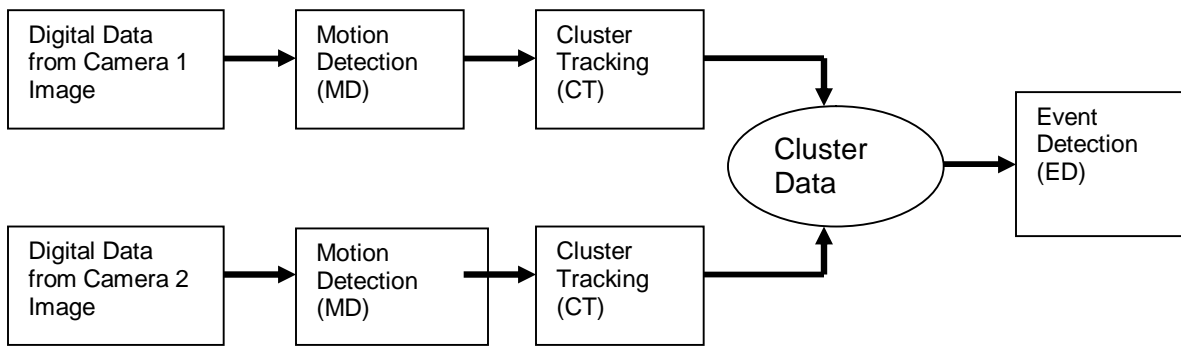


Figure 1 TMDA Block Diagram

3. DETAILED ALGORITHM DESCRIPTION

The following subsections describe in detail the TRISS Motion Detection algorithm:

3.1 Motion Detection Algorithm

This algorithm filters digital information from cameras 1 and 2 to distinguish event data (i.e. explosions) from background clutter. This algorithm performs two separate functions; the first one is to filter the pixel intensity data and the second one is to discriminate event data from background noise as described in more detail in Subsections 3.1.1 and 3.1.2.

3.1.1 Digital Filter

The digitized image data is filtered using a first order lag filter. The block diagram in Figure 2 illustrates the analog representation of the filter using the Laplace Transform.

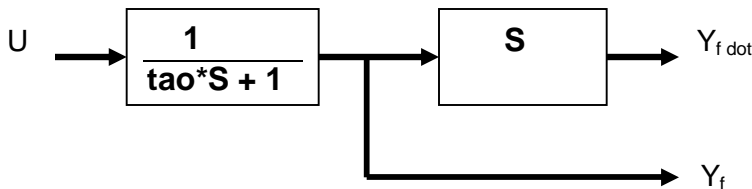


Figure 2 TMDA Filter Block Diagram

Where

- U -- Pixel Intensity (0 –256)
- Y_f -- Filtered pixel intensity
- $Y_{f \dot{}}$ -- Filtered pixel intensity rate
- τ = Filter time constant in seconds

Equations 3-2 and 3-3 depict the discrete Z-Transform representation of the filter using the Tustin (i.e. Trapezoidal) Approximation shown in Equation 3-1.

$$s = (2/dt) * (1 - Z^{-1}) / (1 + Z^{-1}) \quad (\text{Eq. 3-1})$$

$$Y_f = K_1 * Y_f Z^{-1} + K_3 * (U + U Z^{-1}) \quad (\text{Eq. 3-2})$$

$$Y_{f \dot{}} = K_1 * Y_{f \dot{}} Z^{-1} + K_2 * (Y_f + Y_f Z^{-1}) \quad (\text{Eq. 3-3})$$

where:

$$K_1 = (2 * \text{tao} - dt) / (2 * \text{tao} + dt)$$

$$K_2 = 2 / (2 * \text{tao} + dt)$$

$$K_3 = dt / (2 * \text{tao} + dt)$$

dt = Sampling Time in seconds
tao = Filter time constant in seconds

3.1.1 Adaptive Threshold Limits

The rate of change of the pixel intensity, $Y_{f \dot{}}$, is then tested against adaptive lower and upper pixel intensity rate limits as depicted in Figure 3. If the rate of change of the pixel intensity falls outside the upper and lower limit window for a pre-defined number of frames, the pixel is qualified as a “candidate event pixel”. The top diagram is a time plot of the intensity value of one image pixel and its filtered value. The bottom picture is a time plot of the estimated pixel intensity rate of change and the values of the adaptive upper and lower limits.

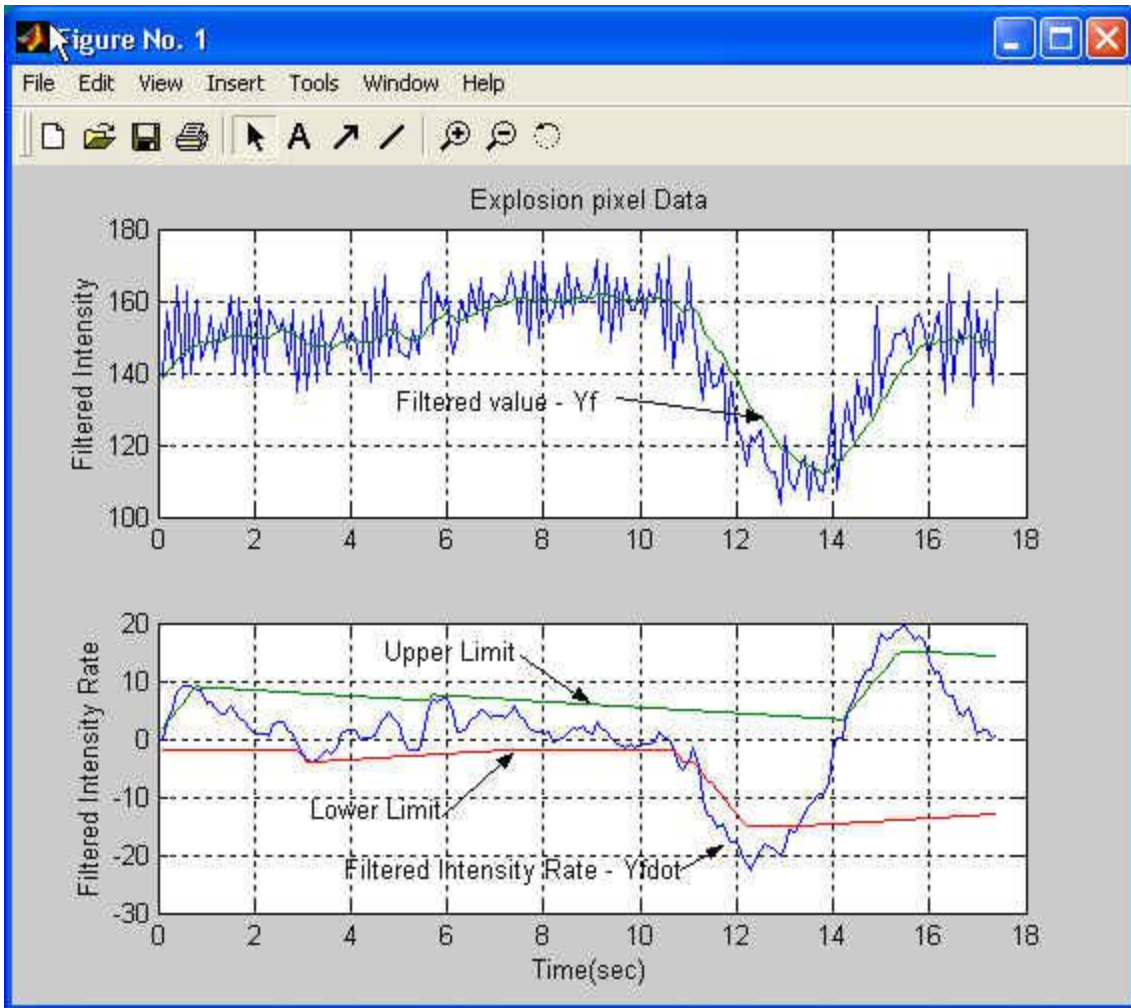


Figure 3 - Plots of Pixel Intensity filtered data

3.2 Cluster Detection Algorithm

This algorithm processes the candidate event data detected by the Motion Detection algorithm to identify and track the centers of multiple event clusters. The pseudo-code for this algorithm is presented in Figure 4.

```

WHILE SEARCH = ON

    LOOP Over Candidate Pixels

        IF Pixel has not been selected THEN
            Determine pixel distance to Search Center
            IF Pixel inside Search Circle THEN
                Recompute Search Circle center
                Tag pixel as Selected
                Increment # of pixels inside Search Circle
            ENDIF
        ENDIF

    END

    IF # Pixels inside search circle > Minimum Number of Pixels THEN
        Found Cluster, Search for more pixels inside this cluster
        Increase Search Circle Radius
        Increase Min Num of Pixels required to keep searching in this cluster
        SEARCH = ON

    ELSE Cluster Search is complete
        IF # Clusters < Maximum Number of Clusters THEN
            Increment # Clusters found
            Save cluster data such as center of cluster and number of pixels

            IF there are more pixels not accounted for THEN
                Continue Search for a new Cluster
                SEARCH = ON
                Reset Search Circle Radius
                Reset Min Num of pixels required to start a cluster
                Select the first non-tagged pixel found the center for a new search
            ELSE
                SEARCH = OFF Search is completed, no more clusters
            ENDIF
        ELSE
            SEARCH = OFF, Complete search reached max number of clusters
        ENDIF

    ELSE
        SEARCH = OFF, Not enough pixels to continue search
    ENDIF
END WHILE

```

Figure 4 Pseudo-Code for the Cluster Detection and Tracking Algorithm

Figure 5 illustrates MATLAB plots of the cluster algorithm tracking data from two real munitions explosions conducted at a Canadian Military Range. At the left bottom corner of the plot, data statistics are shown for frame #102 indicating the number of pixels analyzed, the number of cluster searches, and the corresponding search radiuses for both clusters.

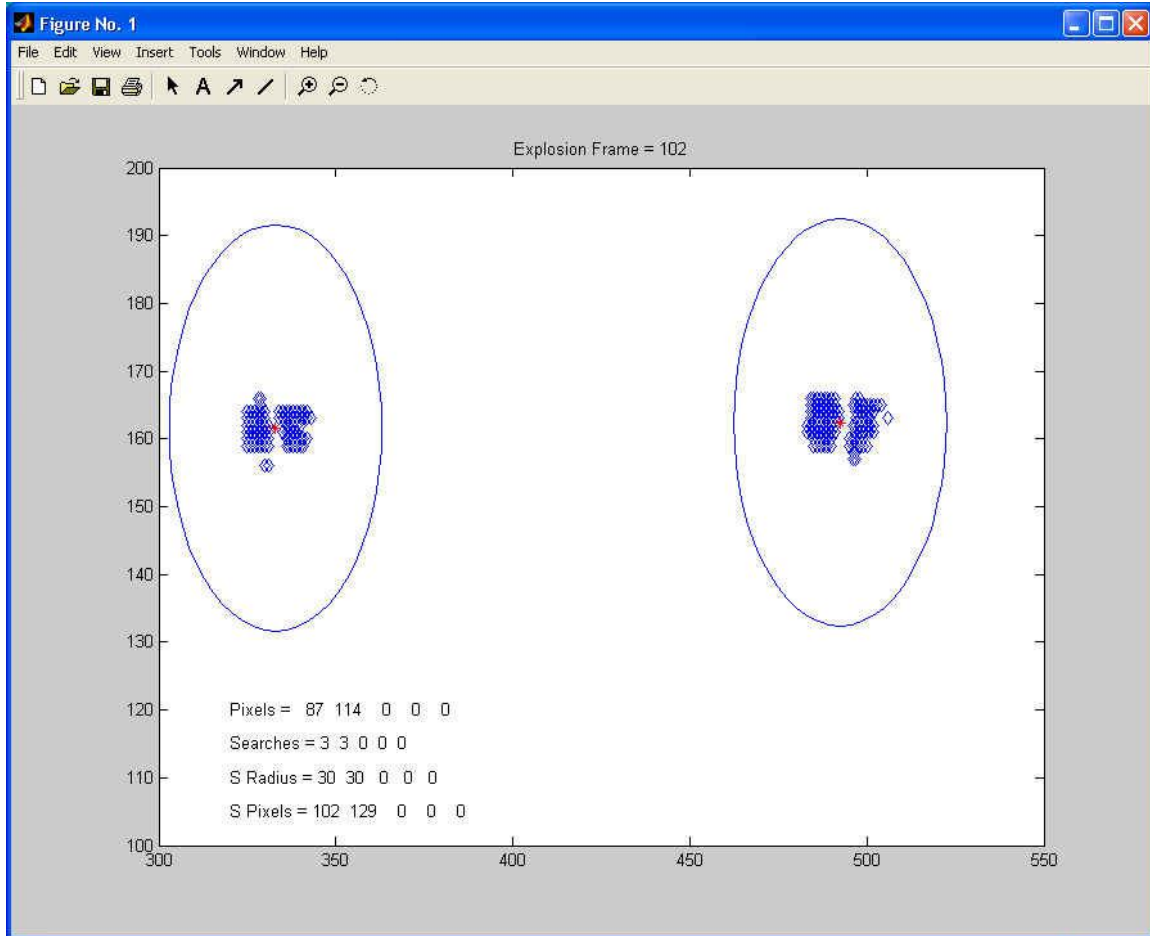


Figure 5 Cluster Detection Algorithm tracking two real events

Figure 6 shows actual TRISS images from camera 1 (top) and camera 2 (bottom) during the detection of an ordnance explosion. The white circle with the letter “S” indicates the position where the MDA algorithm detected the bomb explosion. The white circle with the letter “T” shows the target location.

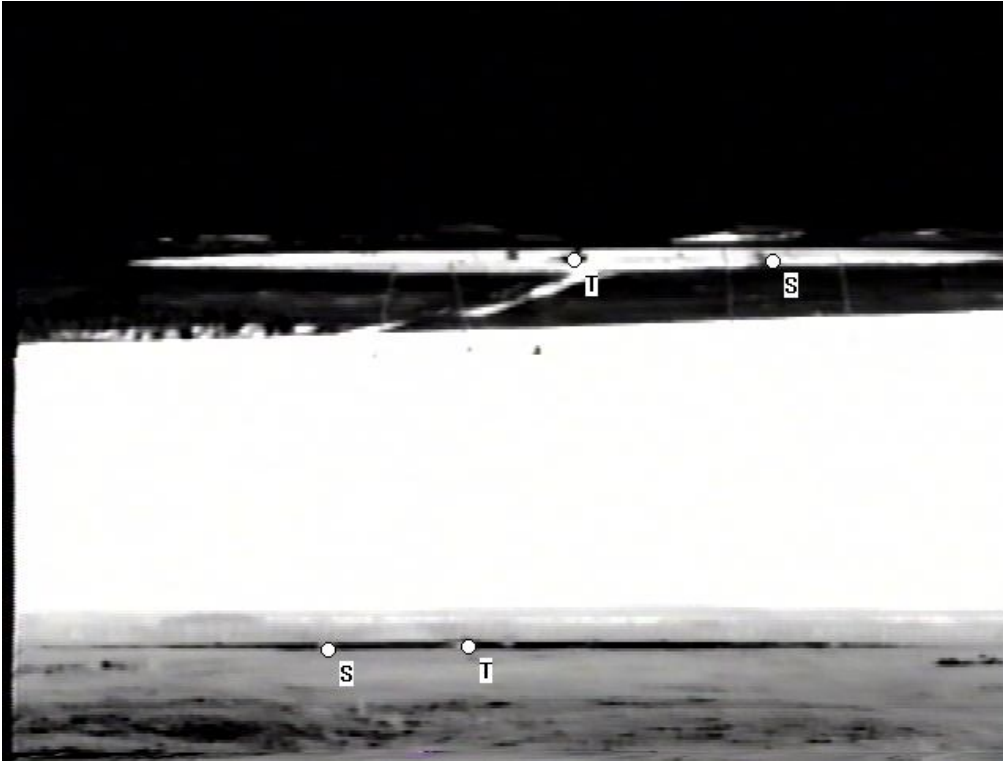


Figure 6 TRISS Images

3.3 Event Detection Algorithm

This algorithm qualifies the event clusters detected by both Cameras by correlating the times when the clusters were detected by cameras 1 and 2. Both cameras must detect the munitions explosion clusters within $1/3^{\text{rd}}$ of a second. A separate TRISS scoring algorithm compares the cluster detected image position with the image coordinates of the selected stationary target for final scoring.