

Demo: Real-time Optical Flow-Based Motion Tracking

Jianwei Zhou and Kefeng Lu, TAMU
Course Instructor: Professor Deepa Kundur

Introduction

The objective of this project is to identify and track a moving object within a video sequence. The tracking of the object is based on optical flows among video frames in contrast to image background-based detection. The proposed optical flow method is straightforward and easier to implement and we assert has better performance. The project consist of two parts, software simulation on Simulink and hardware implementation on TI TMS320DM6437 DSP board. The idea of this project is derived from the tracking section of the demos listed in MATLAB computer vision toolbox website.

The Simulink model for this project mainly consists of three parts, which are “Velocity Estimation”, “Velocity Threshold Calculation” and “Object Boundary Box Determination”. For the velocity estimation, we use the optical flow block in the Simulink built in library. The optical flow block reads image intensity value and estimate the velocity of object motion using either the Horn-Schunck or the Lucas-Kanade. The velocity estimation can be either between two images or between current frame and Nth frame back. We set N to be one in our model.

After we obtain the velocity from the Optical Flow block, we need to calculate the velocity threshold in order to determine what is the minimum velocity magnitude corresponding to a moving object. To obtain this velocity threshold, we first pass the velocity through couple mean blocks and get the mean velocity value across frame and across time. After that, we do a comparison of the input velocity with mean velocity value. If the input velocity is greater than the mean value, it will be mapped to one and zero otherwise. The output of this comparison becomes a threshold intensity matrix, and we further pass this matrix to a median filter block and closing block to remove noise.

After we segment the moving object from the background of the image, we pass it to the blob analysis block in order to obtain the boundary box for the object and the corresponding box area. The blob analysis block in Simulink is very similar to the “regionprops” function in MATLAB. They both measure a set of properties for each connected object in an image file. The properties include area, centroid, bounding box, major and minor axis, orientation and so on. In this project, we utilize the area and bound box measurement. In our model, we only display boundary box that is greater than a

certain size, and the size is determined according to the object to be track. The rest of the Simulink model should be self-explanatory and the details of the Simulink model will be explained in the next section.

Design and Implementation

In this Simulink model, there are couple of major parameters that we need to adjust depending what the tracking object is. The first parameter is the gain after the mean blocks in the velocity threshold subsystem. If too much background noise besides the moving objects is included in the output intensity matrix, the gain need to be adjust to filter out background in the image. The second parameter is the constant that is used for comparison with the boundary box. Any boundary boxes with area below this constant is filter out. One of the disadvantages of optical flow based tracking is that a moving object may have many small boundary boxes due to the optical detection on different part of the moving object. In order to better keep track of the moving object, we need to filter out the small boundary boxes and keep the large boundary box. The other minor parameters such as the shape for the display of motion vector and tracking box are up for the users to decide.

Building the Simulink Model

1. Add a source “From multimedia File” and a sink “To Video Display”. Connect these two blocks to display the input video. The input videos that we used are “optic tracker plane spotting.avi” and “leopard.avi”, and they are included in the submission material.
2. Convert the input video color image to grayscale using “Color Space Conversion”. Set the conversion to “R’G’B’” to intensity”.
3. Pass the grayscale intensity to “Optical Flow” block. In the compute optical flow between tab, select “Current frame and Nth frame back” and set $N = 1$. In velocity output tab, select “in horizontal and vertical components in complex form”. We choose to output velocity in complex form because we would like to display the motion vector on top of the original image.
4. Pass the complex velocity vector to “Optical Flow Line” block. In the gain tap set the gain to be 10. The optical flow line block generates the coordinate for the input velocity vector.
5. Add a “Draw Shapes” block. In shape tap, select “line”. Connect the original video to the image input of draw shapes block, and connect optical flow line block to Pts input of draw shapes block. Connect the output of draw shapes block to a video display block.

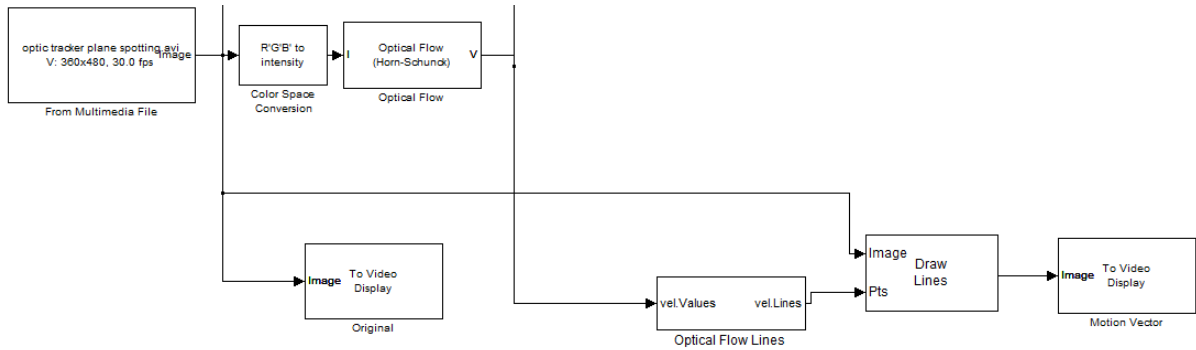


Figure 1: Part of Simulink Model to be implemented

6. Create a subsystem to determine velocity threshold.
 - a. First set the input port of this subsystem to be the velocity output from the optical flow block.
 - b. Pass the velocity to a “Math Function” block and set the function be “ magnitude^2 ”.
 - c. Pass the velocity magnitude square to two “Mean” blocks. For the first mean block, select “Find the mean over Entire Input”. For the second mean block, select “Running Mean”. The output of these two mean blocks returns the mean velocity vector across frame and time. A gain block might be necessary after the mean blocks in order to better determine the velocity threshold.
 - d. Add a “Operational Operator” block and select “ \geq ”. This operator map input velocity to one if it is greater than the mean velocity and one otherwise.
 - e. Add a “Median Filter” block.
 - f. Add a “Closing” block.

The output of this subsystem is an intensity matrix that represents the moving object that we want to track.

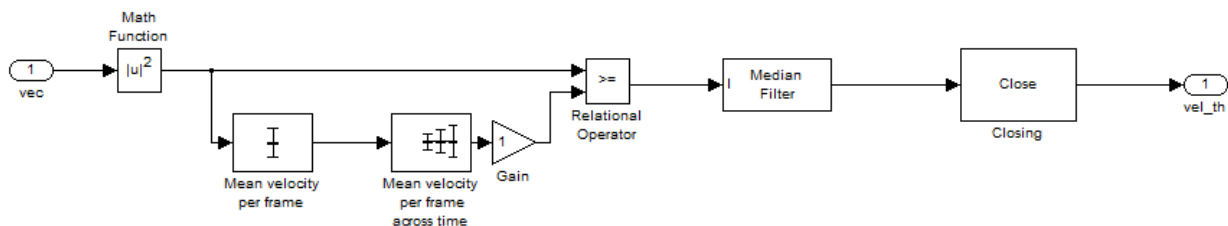


Figure 2: Subsystem that determines velocity threshold and segment the moving object

7. Create a subsystem to determine the boundary box of the moving object.
 - a. Set the intensity matrix from the previous subsystem as input port
 - b. Add a “Blob Analysis” block and select “Area” and “BBox” as block outputs.

- c. Add a “Compare To Constant” block and set the constant to be 3000. The value of this constant determines the minimum boundary box size to be displayed. The value should be adjusted according to the object to be tracked.
- d. Add a “Submatrix” block and use default bock setting.
- e. Connect the intensity matrix to BW input port, Area output port to compare to constant block, BBox output to submatrix block.
- f. Add a “logical operator” and set the operator to be “AND”. Connect the compare to constant block and submatrix block to AND operator. The output of this AND operator provides the index of boundary box that above the constant we set previously.
- g. Add a “Variable Selector” and choose “Column” in the select tap.
- h. Pass the BBOX to In1 of variable selector block, and output of AND operator to idx of variable selector block.

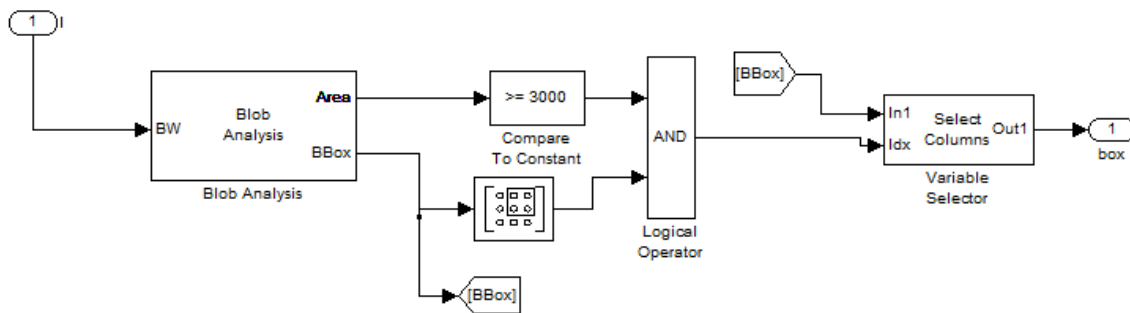


Figure 3: Subsystem that determines the boundary box of the moving object

- 8. Add a “Draw Shapes” Block. Select the shape to be rectangle. Connect the original video and the boundary box to the draw shape block.

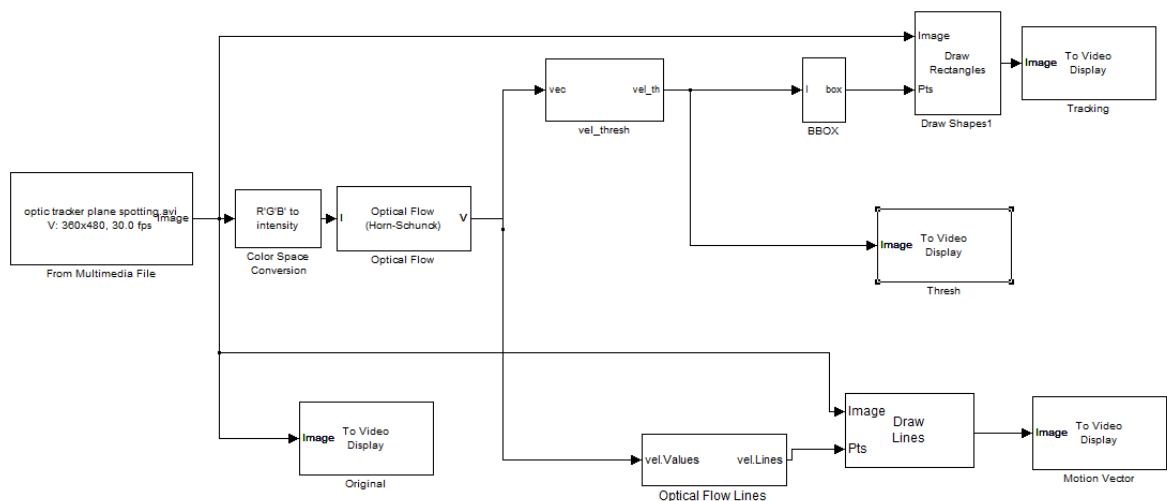


Figure 4: Complete Optical Flow Based Motion Tracking Simulink Model

Digital Video Hardware Implementation

1. Go to the Simulation → Configuration Parameters, change the solver options type back to “Fixed step”, solver to “discrete (no continuous states).”
2. Start with your Simulink Model for Optical Flow Based Motion Tracking and delete all the input and output blocks. Besides, to release the burden of the hardware, we only retain blocks related to the final tracking video and delete all other unnecessary blocks.
3. Under Target Support Package → Supported Processors → TI C6000→Board Support→DM6437EVM, found “Video Capture” block. Set Sample Time to -1.
4. Add a “Video Display” block, and set Video Window to “Video 0”.
5. Add a “Deinterleave” block and an “Interleave” block from DM6437 EVM Board Support. Link the “Video Capture” block to the “Deinterleave” block. And Link the “Interleave” block with a Video Display.
6. Add an “Image Data Type Conversion” block from the Video and Image Processing Blockset → Conversions. Set the output data type to “uint 8”, and link its input with the output of the “Draw Rectangles” Block. Add another two “Data Type Conversion” blocks from the Video and Image Processing Blockset → Conversions. Set the output data type to “single”. Link one block between the “Deinterleave” block and the “Optical Flow” block, and link another block between the “Deinterleave” block and the “Image” input of the “Draw Rectangles” Block.
7. Link Cb and Cr channels between “Deinterleave” block and both of the “Interleave” blocks, as shown in Figure 5.
8. Adding a DM6437EVM Block under Target Support Package TI C6000→Target Preferences.
9. Connect the video output (the yellow port) of the camera with the video input of the board, and the output of the board with the monitor.
10. Go to Tools->Real Time Workshop -> Build the model.

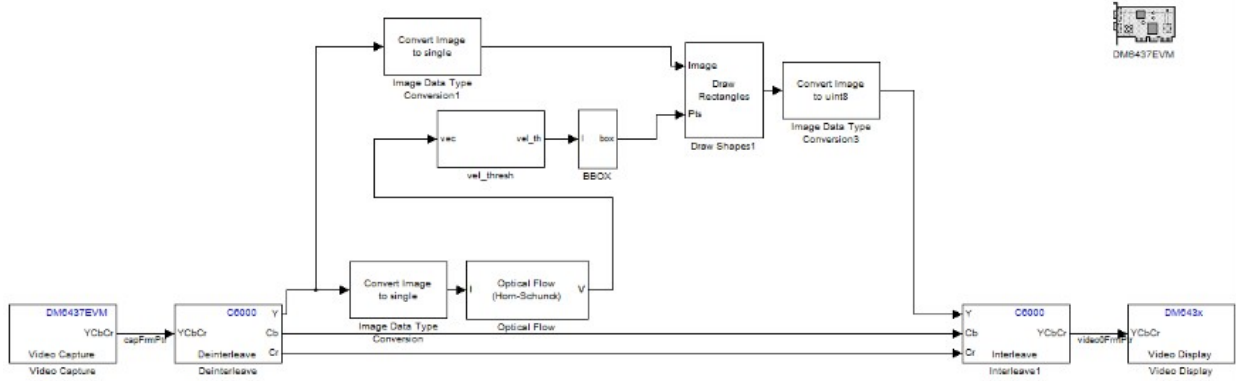


Figure 5: Complete Optical Flow Based Motion Tracking Hardware model.

Results

Simulink Simulation

Input video file: optic tracker plane spotting.avi

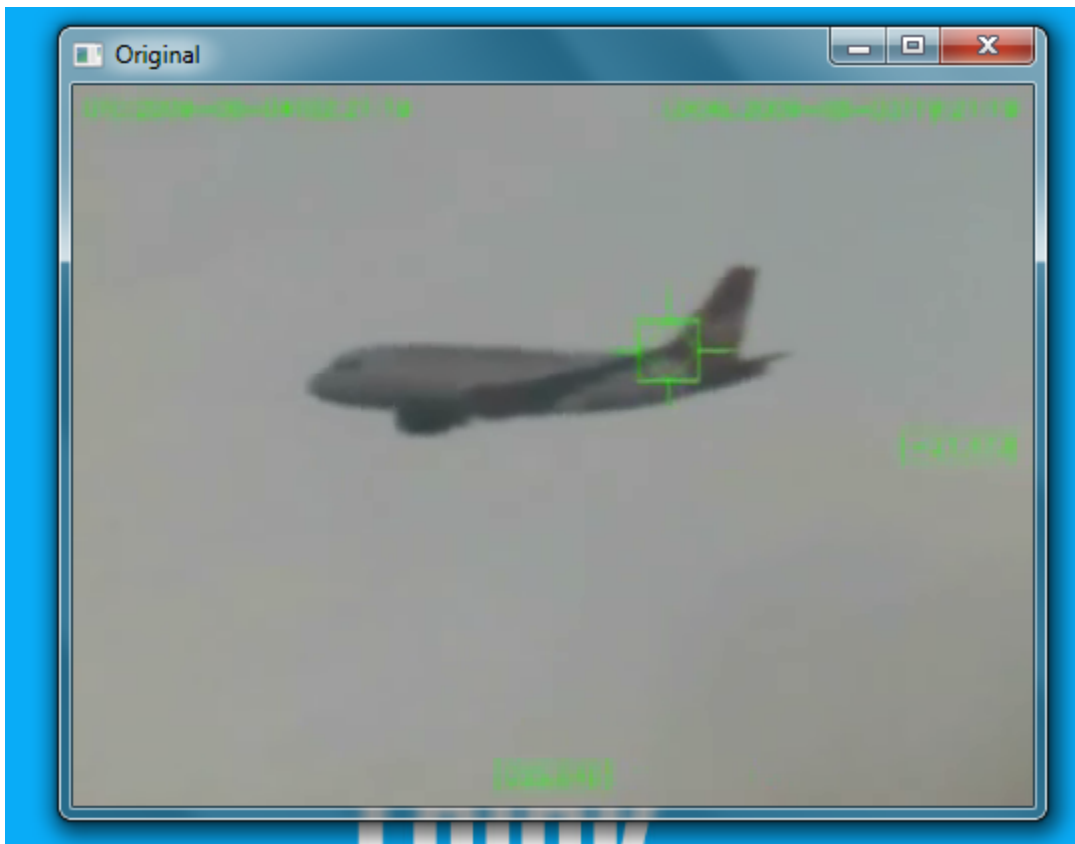


Figure 6: Original input video

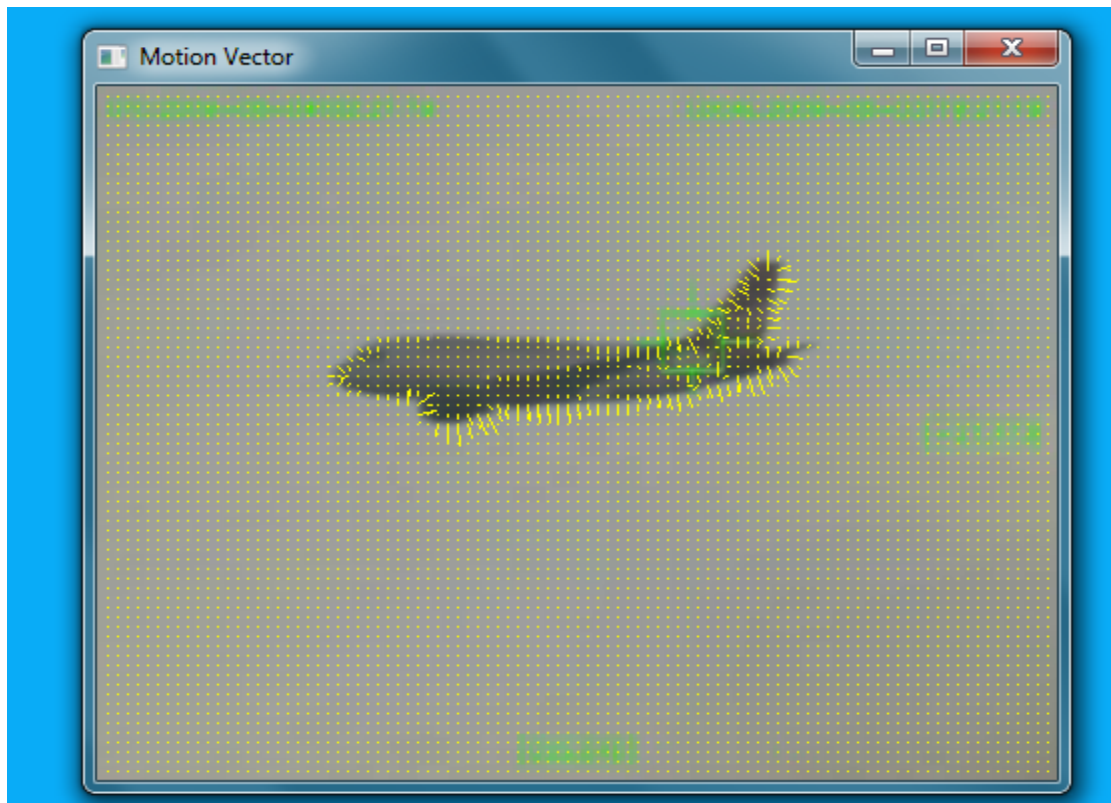


Figure 7: Video with motion vector

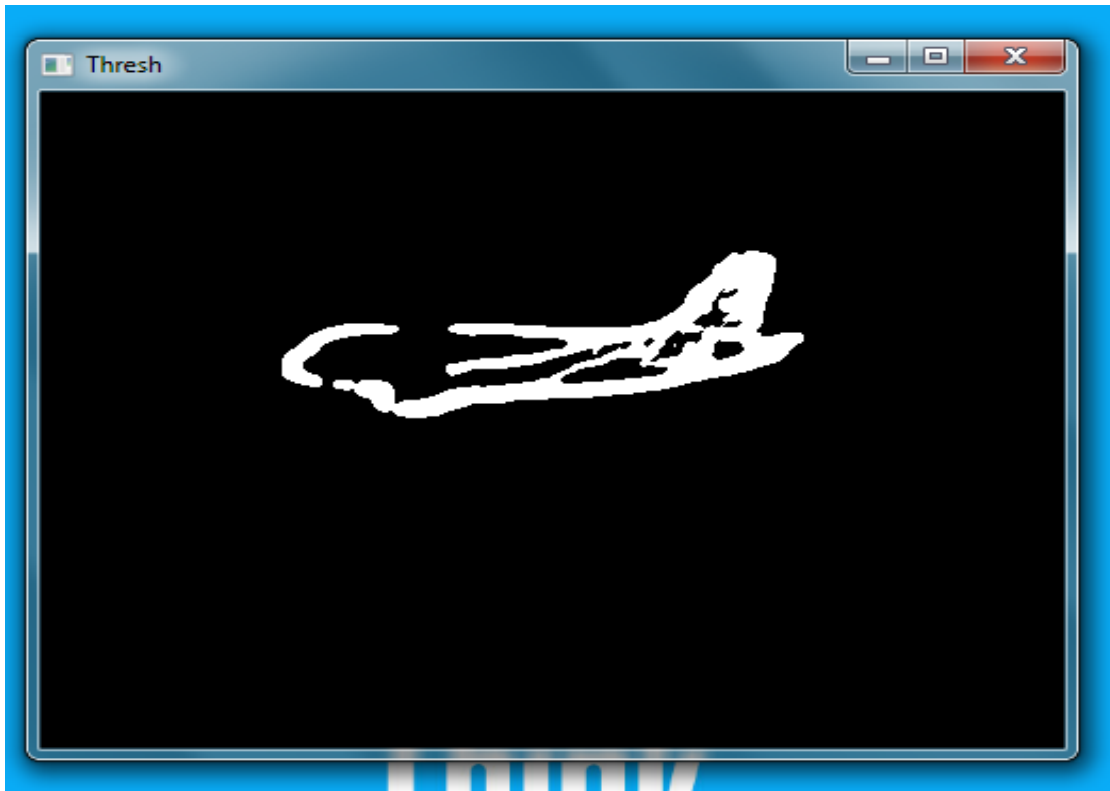


Figure 8: Video of the moving object after segmentation



Figure 9: Video of moving object tracking by a box

Input video file: leopard.avi

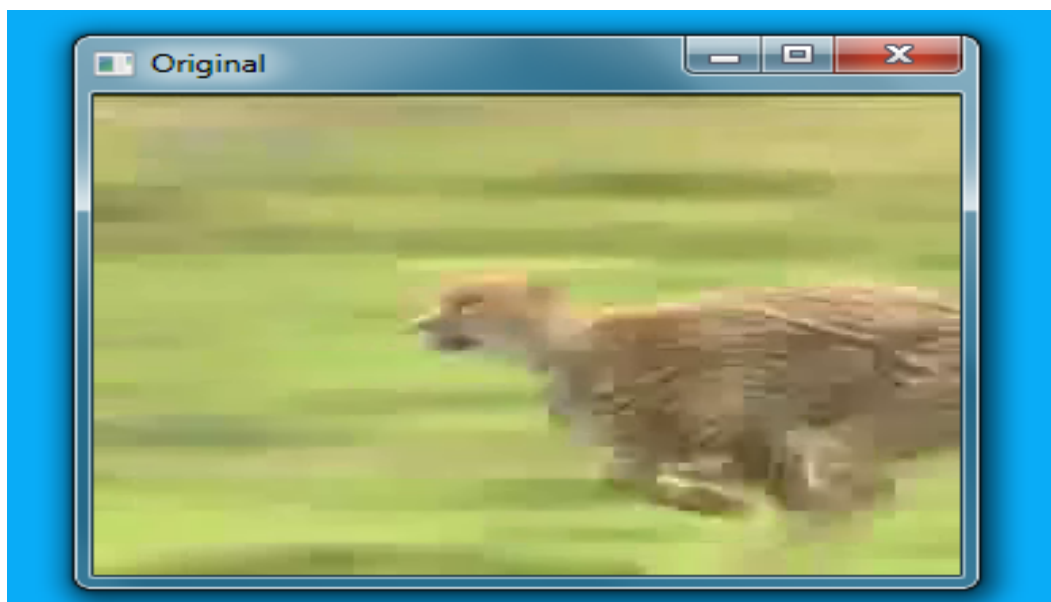


Figure 10: Original input video

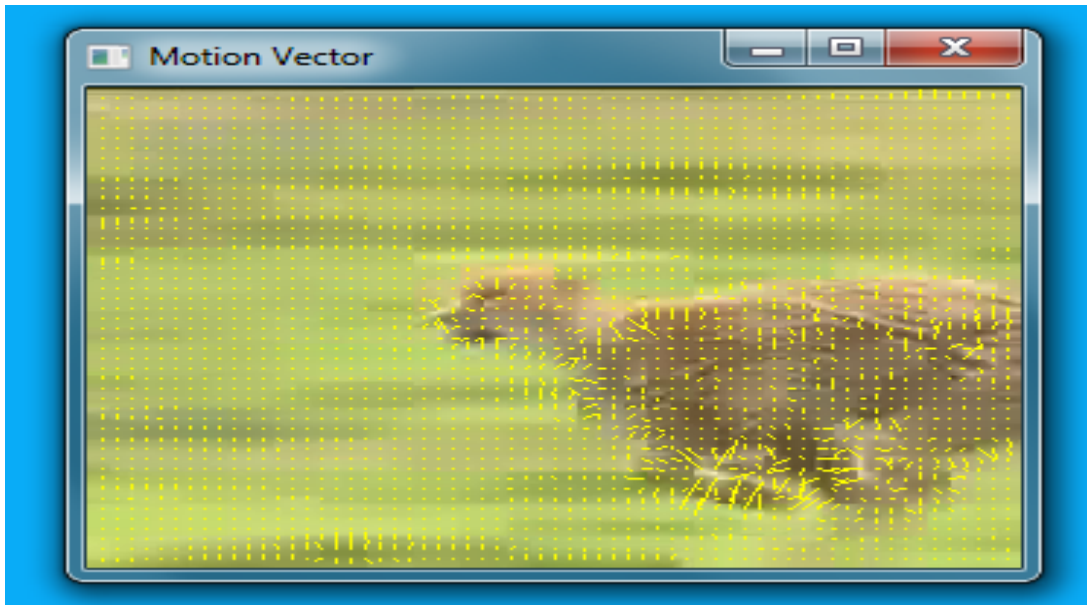


Figure 11: Video with motion vector

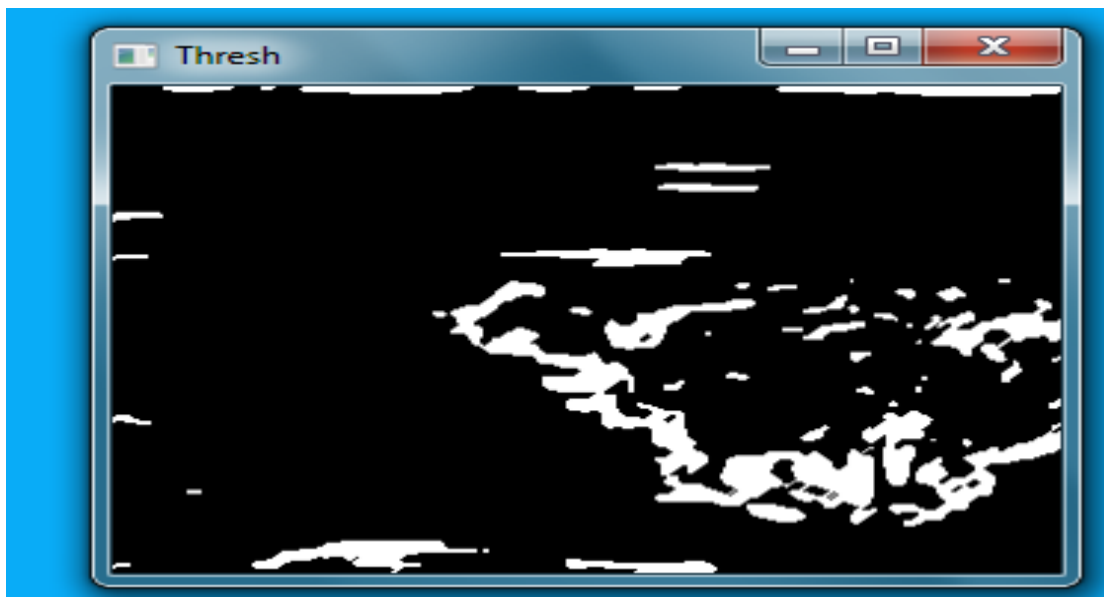


Figure 12: Video of the moving object after segmentation

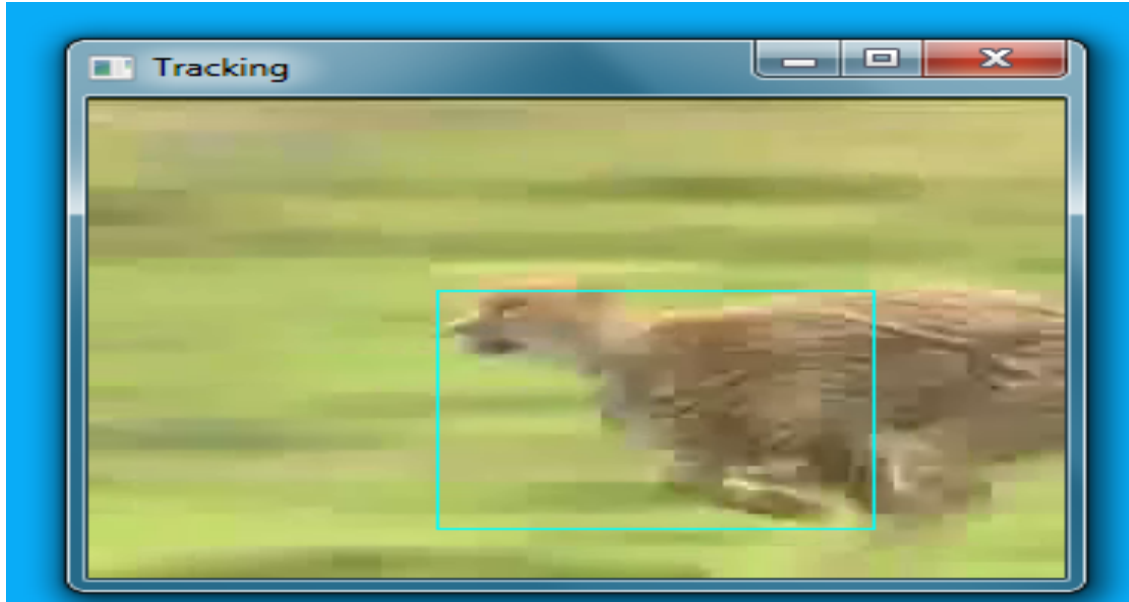


Figure 13: Video of moving object tracking by a box

We run our Simulink model with two different videos, one for a flying airplane and one for a running leopard. Therefore, the tracking objects are airplane and leopard respectively. From our simulation observation, we notice that our Simulink Model can track moving object with approximately same speed in a constant direction very well. It might take a short time before the model can identify the moving object. The tracking box might not be able to cover the whole object, but the tracking ability of this model can still be met. If the background of the moving object is very complicated, it might affect the tracking performance. Depending on the background, the model may lose track of the object and re-track the object again. As explain in section two, the gain in the velocity threshold subsystem and the constant in the boundary box subsystem need to be adjusted according case by case. In our airplane and leopard examples, we set the gain to be 1 and the constant to be 2000. These parameters are the same because both the airplane and the leopard in the videos exhibit similar kind of motion.

Hardware Implementation Result

We implement the model on TI TMS320DM6437 DSP board. And the results are as follows:

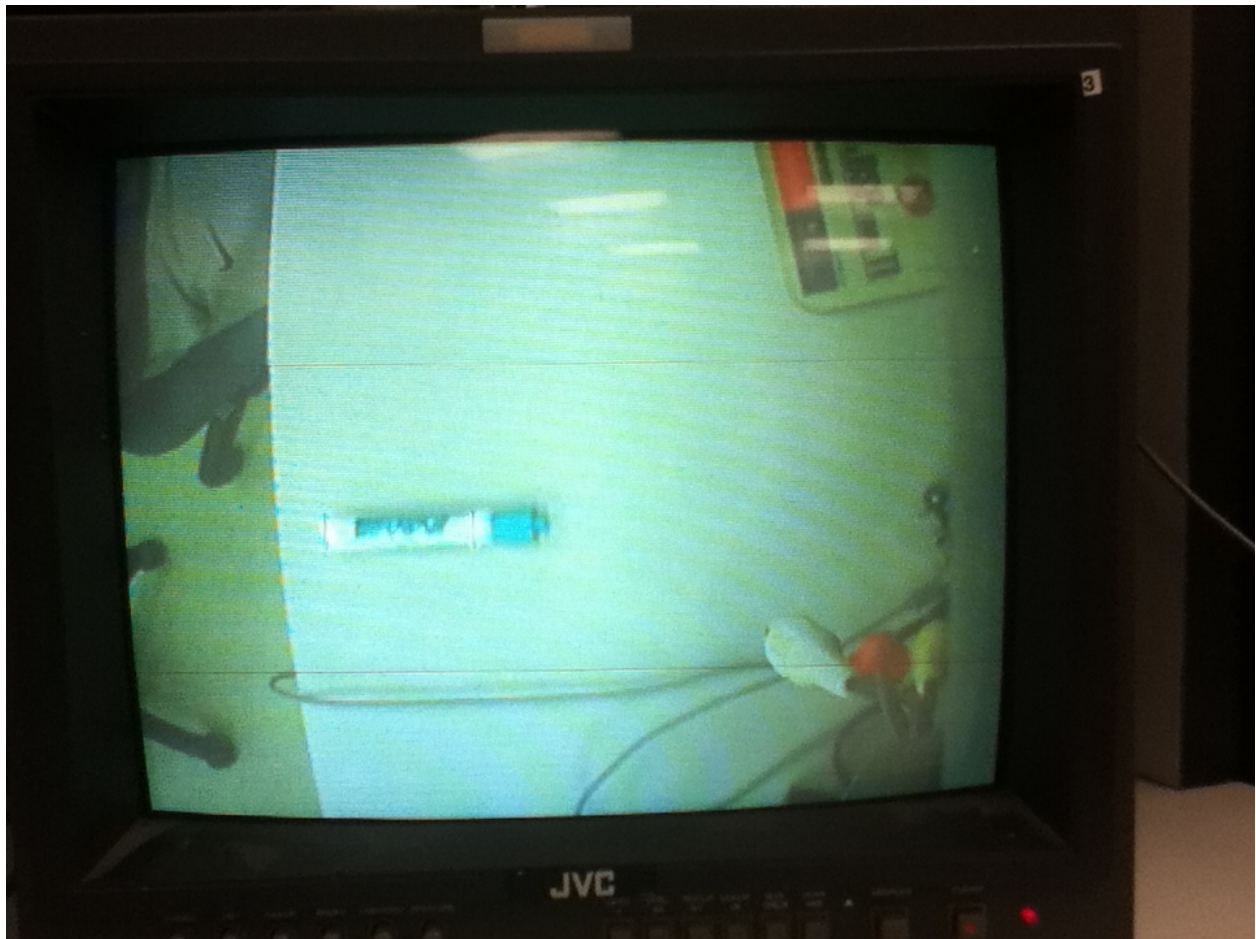


Figure 14-1: Video of moving object tracking by a box

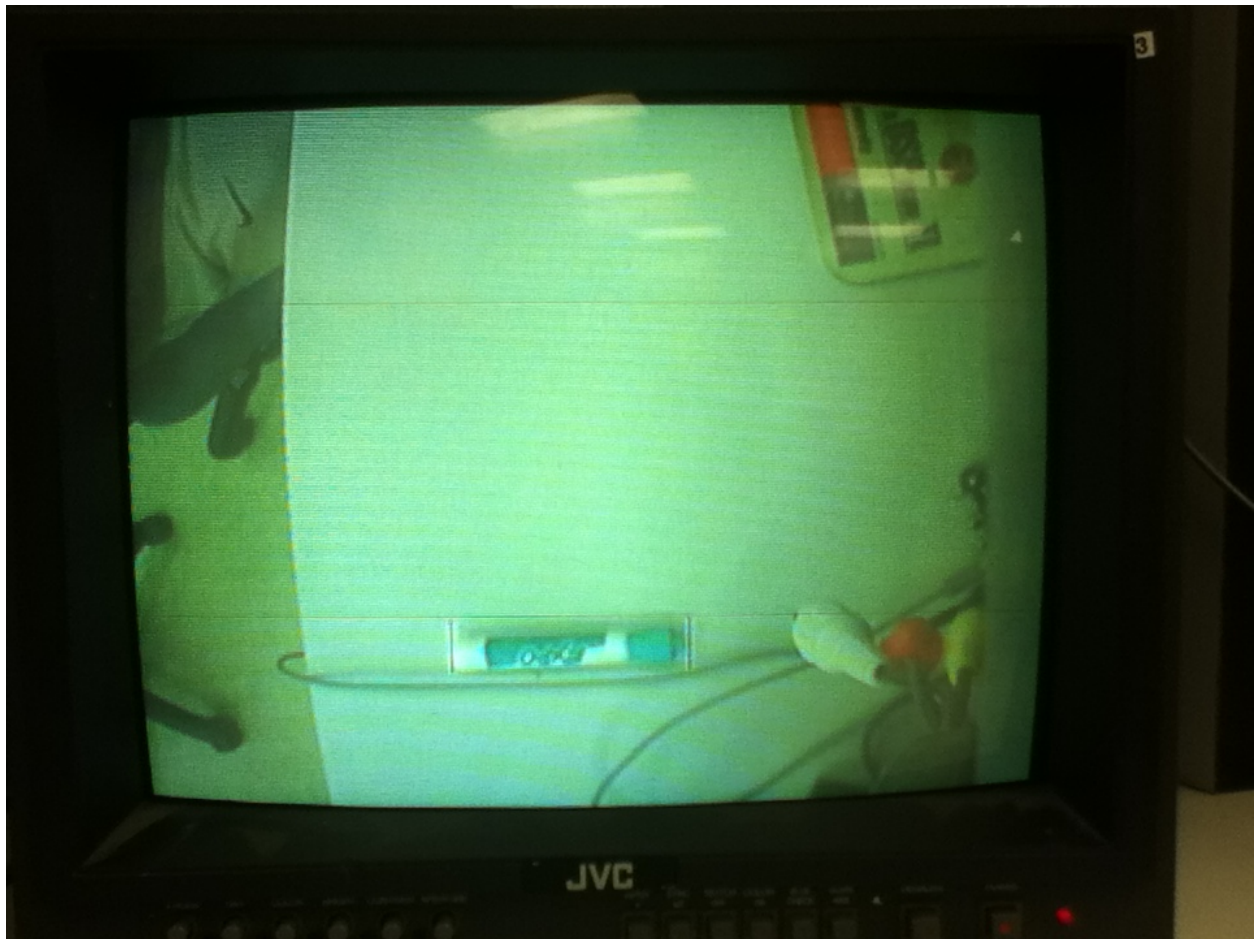


Figure 14-2: Video of moving object tracking by a box

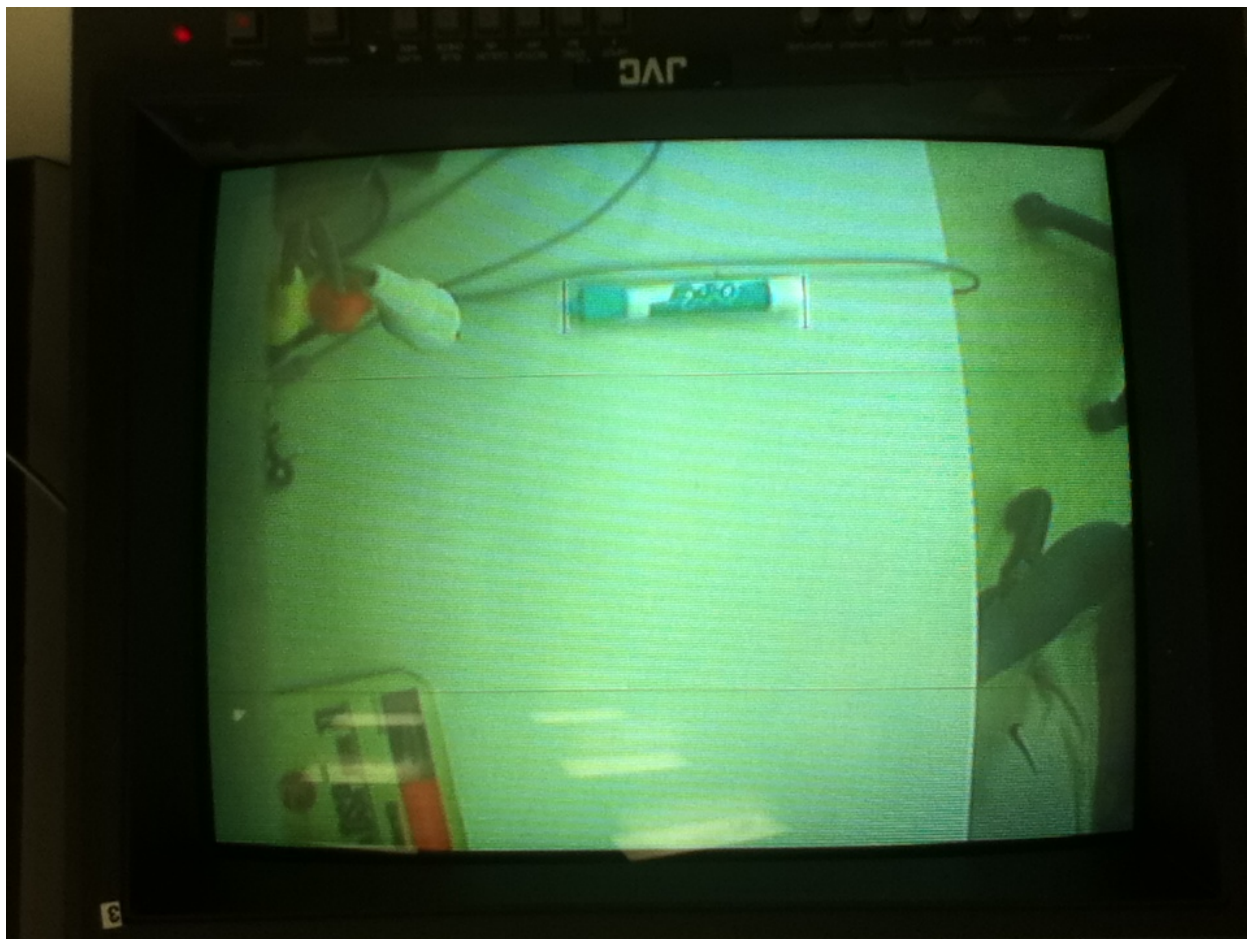


Figure 14-3: Video of moving object tracking by a box

As we can see the pictures above, we use a black rectangle to tracking the moving mark pen. Since the DSP board only supports two output video windows at the same time, we delete all the unnecessary outputs except the final tracking output. The reason that we remove the original window is that we can only add a black rectangle on it, so we can see the original video at the final video. There are two major parameters that we need to adjust depending what the tracking object is. The first parameter is the gain after the mean blocks in the velocity threshold subsystem. The second parameter is the constant that is used for comparison with the boundary box. Any boundary boxes with area below this constant is filter out. In order to better keep track of the moving object, we need to filter out the small boundary boxes and keep the large boundary box. We adjust the gain parameter to a small amount 0.5, since the background is stay still during the experiment and there is little change in the video except the moving object. Besides, we set the area parameter to a medium amount 1500, because the size of mark pen is in a medium size. If we set this parameter to a relative small amount such as 200, we will see many small rectangles in the video which is not desired. We want to track the integrated object in the video, not separate the object into many small pieces.

The main difficulty that we ran into in the testing time is the slow processing speed of TI TMS320DM6437 DSP board. The board needs a few seconds to process the image data in a single frame, so we can only see the separate images with a several seconds interval. Fortunately, we can still see the tracking effect of the moving object in the video. If we want to do this tracking to a real time level, we need a more powerful DSP board to implement the model.

References

"Computer Vision System Toolbox -Demos and Seminar Tracking Section",
mathwork.com, n.p., n.d, Mon. 30 Nov. 2011