GM8210/GM828x/GM8139/GM8136

MOTION DETECTION

Programming Guide
Rev.: 1.0
Issue Date: October 2014

REVISION HISTORY

GM8210 GM828x GM8139 GM8136 Motion Detection Programming Guide

Date	Rev.	From	To
Oct. 2014	1.0	-	Original

Printed in Taiwan 2014

Grain Media, Inc.
5F, No. 5, Li-Hsin Road III, Hsinchu Science Park, Hsinchu City, Taiwan 300, R.O.C.

Grain Media's home page can be found at:
http://www.grain-media.com

TABLE OF CONTENTS

Chapter 1

Introduction

This chapter contains the following sections:

1.1 Overview

The motion detection in GM8210/GM828x/GM8139/GM8136 is used to detect the moving objects in video and it consists of two parts. The first part, capture IP, provides the background model of the current captured frames and the rough motion detection result that are based on the difference between the background model and captured frames. The second part, motion detection AP, provides the final motion detection result after filtering out noise and provides the user interface for users to control the motion detection application in GM8210/GM828x/GM8139/GM8136. The motion detection has two different AP versions, the first version supports the switch for each block to turn on/off the motion detection and the second version supports the sub-region motion detection.

1.2 Features

The motion detection application contains the following features:

- Supports maximum motion detection resolution of 1920 x 1080
- Supports alarm system when total motion block numbers of frames exceed predefined threshold
- Supports switch for each block to turn on/off motion detection
- Supports sub-region motion detection

Chapter 2

Motion Detection

This chapter contains the following sections:

2.1 Motion Detection Algorithm

The motion detection provides the motion detection result of the current captured frame. The steps of motion detection are listed as follows:

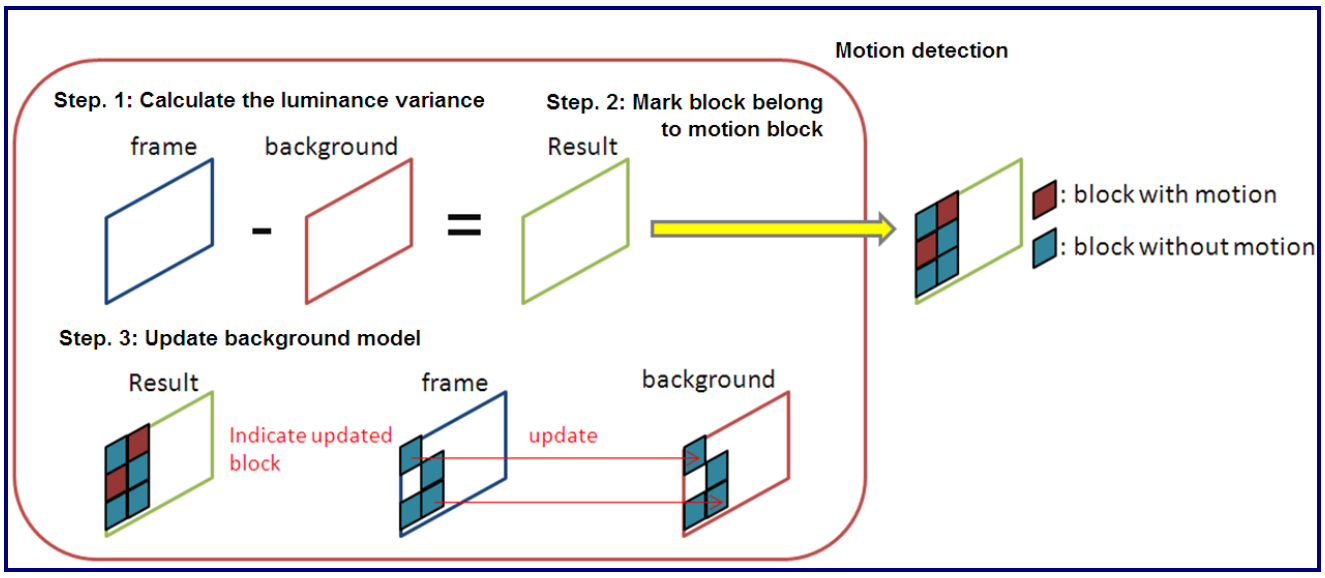
Step 1: Calculate the luminance variance between the current captured frame and background model

Step 2: Mark the block that belongs to the motion block if the luminance variance exceeds the predefined threshold

Step 3: Update the background model using the captured frame

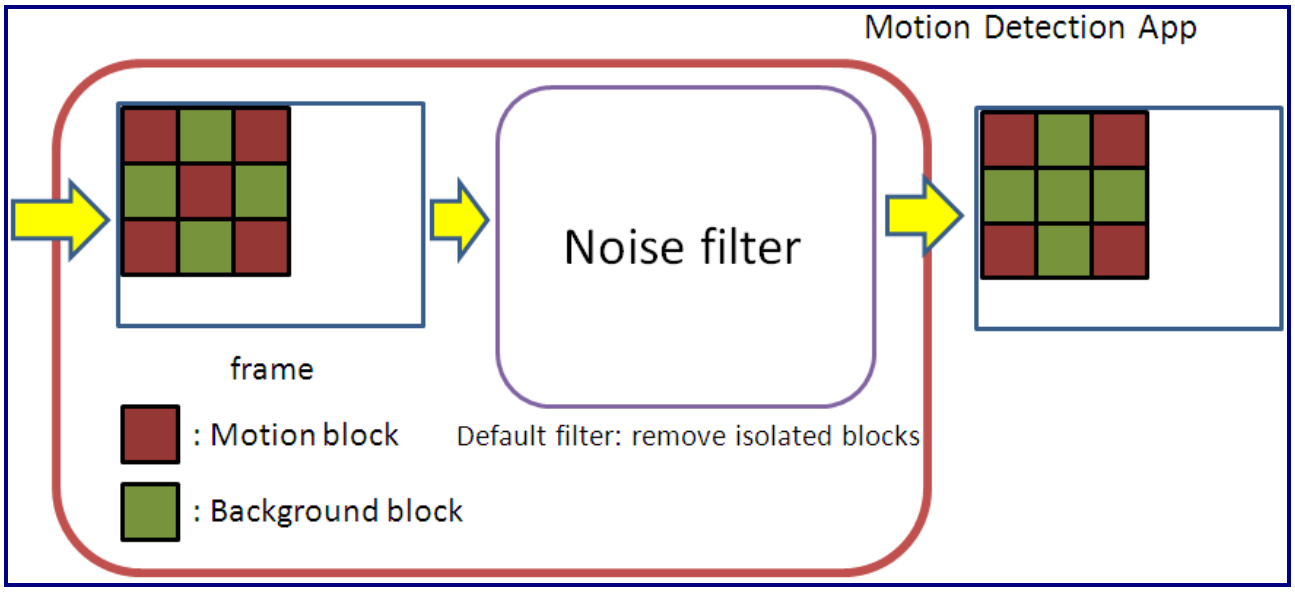
2.2 Motion Detection Flow Chart

The flowchart of the motion detection is shown as below:



2.3 Motion Detection Post-processing

The AP of the motion detection includes a simple noise filter with the isolated motion block as noise and removes it. It can refine the motion detection result. The adopted noise filter can be adjusted by customers depending on the application.



Chapter 3

Data Structure of Application

This chapter contains the following sections:

3.1 Parameter Structure for Motion Detection

```
struct mdt_user_t {
        struct mv_cfg_t  mv_param;
struct mdt_alg_t mdt_alg;
        pthread_mutex_t  mdt_mutex;
void *bindfd;
}
```

Element	Description
struct mv_cfg_t mv_param	Parameter for the motion data
struct mdt_alg_t mdt_alg	Parameter for the motion detection application
pthread_mutex_t mdt_mutex	mutex for the resource protection
void *bindfd	For check only

```
struct mv_cfg_t {
        unsigned int k_height;
        unsigned int k_width;
unsigned int k_mb_height;
unsigned int k_mb_width;
unsigned char *active_flag;
unsigned int active_num;
};
```

Element	Description
unsigned int k_height	Vertical resolution of kernel mv data
unsigned int k_width	Horizontal resolution of kernel mv data
unsigned int k_mb_height	Macroblock vertical resolution of kernel mv data, k_width < 1024, k_mb_height = 16; otherwise, k_mb_height = 32
unsigned int k_mb_width	Macroblock horizontal resolution of kernel mv data, k_width < 1024, k_mb_width = 16; otherwise, k_mb_width = 32
unsigned char *active_flag;	The parsed motion data for algorithm
unsigned int active_num;	The number of macroblocks with motion for algorithm

```
struct mdt_alg_t {
struct mdt_reg_t *sub_region;
unsigned int    sub_region_num;
unsigned int    u_height;
unsigned int    u_width;
unsigned int    u_mb_height;
unsigned int    u_mb_width;
unsigned int    sensitive_th;
unsigned int    frame_count;
unsigned int    training_time;
unsigned int    mb_w_num;
unsigned int    mb_h_num;
};
```

Element		Description
struct mdt_reg_t *sub_region;		Information used for the sub-region motion detection (Only for AP version 2)
unsigned int sub_region_num;		Sub-region number (Only for AP version 2)
unsigned int	u_height;	User-defined height It is suggested not changing the default value.
unsigned int	u_width;	User-defined width It is suggested not changing the default value.
unsigned int	u_mb_height	User-defined mb height (The suggested value = k_mb_widthh * 2) If the value is too large, it would affect the accuracy of the motion detection.
unsigned int	u_mb_width;	User-defined mb width (The suggested value = k_mb_widthh * 2), If the value is too large, would affect the accuracy of the motion detection.
unsigned int	sensitive_th;	Control the transform form the kernel mv data to the user mv data (The suggested value = 60)
unsigned int	frame_count;	Indicate the number of the trained frames
unsigned int	training_time;	Needed number of the training frames for the motion detection application
unsigned int	mb_w_num;	The macroblock number of the width
unsigned int	mb_h_num;	The macroblock number of the height

3.2 Activity Structure for Motion Detection

```
struct md_res{
unsigned int active_num;
unsigned char *active_flag;
} md_res;
```

Element	Description
unsigned int active_num	The number of the activity blocks in the current frame
unsigned char *active_flag	Motion result flag of each MB cell

Chapter 4

Motion Detection APIs

This chapter contains the following sections:

4.1 Introduction

The following table summarizes various functions that are used to call the application interfaces to motion detection.

Function Name	Description
motion_detection_init ()	Initiate the motion detection engine and allocate all resources used internally
set_interesting_area ()	Set the parameter for the motion detection application
set_cap_motion ()	Set the parameter for the capture driver
motion_detection_update ()	Update the setting parameter to the motion detection application
motion_detection_handling ()	Main detection function that detects the current frame while encoding
motion_detection_end ()	Release all resources allocated by the capture motion detection engine

4.2 Basic Function

4.2.1 motion_detection_init

Description	This function should be called for initializing all information parameters used in the motion detection and allocate all resources used internally.
Syntax	motion_detection_init ()
Argument	-
Return	0: Successful 1: Failure

4.2.2 set_interesting_area

Description	This function sets the parameter for the motion detection application
Syntax	set_interesting_area (int ch)
Argument	**int ch** Channel index
Return	0: Successful 1: Failure

4.2.3 set_cap_motion

Description	This function sets the parameter for the capture driver
Syntax	set_cap_motion(int ch, int param, int value)
Argument	**int ch** Channel index **int param** Parameter index **int value** Parameter value
Return	0: Successful 1: Failure

4.2.4 motion_detection_update

Description	This function updates the setting parameter to the motion detection application.
Syntax	motion_detection_update (void * bindfd , struct mdt_alg_t mdt_alg)
Argument	**void * bindfd** Parameter for GM AP **struct mdt_alg_t mdt_alg** Parameter for motion detection
Return	0: Successful 1: Failure

4.2.5 motion_detection_handling

Description	This is the main detection function that detects the current frame while encoding.
Syntax	int motion_detection_handling (gm_multi_cap_md_t *cap_md, struct mdt_result_t *mdt_result, int MAX_CH_NUM)
Argument	**gm_multi_cap_md_t *cap_md** This is a structure pointer to the structure, cap_md and cap_md array store the md event data from capture driver **struct mdt_result_t *mdt_result** This is a structure pointer to the structure, mdt_result. **int MAX_CH_NUM** This parameter indicates the maximum channel number.
Return	0: Successful 1: Failure

4.2.6 motion_detection_end

Description	This function should be called to release all resources allocated by the capture motion detection Engine.
Syntax	int motion_detection_end ()
Argument	-
Return	0: Successful 1: Failure

Chapter 5

Example Code

This chapter contains the following section:

- 5.1 Main File

5.1 Main File

This sample performs the capture motion detection on GM8210/GM828x/GM8139/GM8136. In this example, it receives md_event data and detects each frame using md_event information.

```
static int set_interesting_area(int ch)
{
   ret = motion_detection_update(bindfd[ch], &mdt_alg);
   return ret;
}

static void *motion_thread(void *arg)
{
   while (enc_exit == 0) {
      ret = gm_recv_multi_cap_md(cap_md, MAX_CH_NUM);

               ret = motion_detection_handling(cap_md, &mdt_result[0], MAX_CH_NUM);

      for (ch = 0; ch < MAX_CAP_MD_NUM; ch++) {

         if (mdt_result[ch].result == MOTION_DETECTED)
             printf("[---Left Area Has Motion Detected---] at CH(%d)\n", ch);
      }
      usleep(200000);   //two second period to detect motion
   }
   return 0;
}

int main(void)
{
    motion_detection_init();//motion detection initial

    ret = set_interesting_area(ch);

    ret = pthread_create(&motion_thread_id, NULL, motion_thread, (void *) NULL);

    return 0;
}
```

Chapter 6

Parameters Setting

This chapter contains the following sections:

6.1 AP Parameter

The encode_with_capture_motion_detection sample code provides the mb_cell_en parameter to turn on/off the motion detection of the blocks and the alarm_th parameter to control the motion alarm threshold.

The encode_with_capture_motion_detection2 sample code provides the sub_region parameter to turn on/off the motion detection of the sub-region and the alarm_th parameter to control the motion alarm threshold.

sensitive_th in both sample codes is used to control the transform between the motion data in the kernel layer and the user layer and it is suggested not adjusting this parameter.

6.2 Capture Parameter

The capture parameters that control the motion detection result are listed as follows:

1. alpha: This parameter controls the background model learning rate, if the variance of the background luminance is obvious which might be caused by the sun light or AE, users can select high value for alpha to avoid the false alarm caused by the luminance variance.
2. tbg: This parameter is the threshold of the background model weight. It is suggested not adjusting this parameter.
3. init_val: This parameter is the initial value for the background model. It is suggested not adjusting this parameter.
4. tb: This parameter is the threshold parameter which decides the block belonging to the foreground or not.
5. sigma: This parameter controls the variance of the background model, if the variance of the background luminance is obvious, users can select high value for sigma to avoid the false alarm caused by the luminance variance.
6. alpha accuracy: This parameter controls the updated speed between the background models. The alpha accuracy with large value can reduce the training time of the background model, but it would cause the background model unstable.
7. tg: This parameter is the threshold parameter which decides the block belonging to the background or not.