# MI MD
# User Manual

V1.4

# REVISION HISTORY

| Revision No. | Description | Date |
|---|---|---|
| 1.0 | • Created. | 03/22/2017 |
| 1.1 | • Return value integration. | 04/27/2017 |
| 1.2 | • Add recommendation for width of region/window | 06/07/2017 |
| 1.3 | • Modify the return value error of MI_MD_GetWindowParamsIn<br>• Increase the number of divisions of window | 07/20/2017 |
| 1.4 | • Add MI_MD_ComputeImageSAD API | 09/15/2017 |

# TABLE OF CONTENTS

# 1. INTRODUCTION

## 1.1. Purpose

Motion detection is a function used for detecting moving objects in the scene captured by video camera. The applications for this function are, for example, security monitoring, or motion-trigger recording to save storage space.

# 2. API REFERENCE

## 2.1. API Overview

MI_MD_Init: Initialize MD library.
MI_MD_Uninit: Release memory of library.
MI_MD_Run: Run MD library.
MI_MD_SetRegionInfo: Set parameters for independent region motion detection.
MI_MD_SetDetectWindow: Set motion detection window position.
MI_MD_GetDetectWindowSize: Get motion detection window position.
MI_MD_SetWindowParamsIn: Set parameters for certain motion detection sub-window.
MI_MD_GetWindowParamsIn: Get parameters for certain motion detection sub-window.
MI_MD_GetWindowParamsOut: Get result for certain motion detection sub-window.
MI_MD_SetTime: Set the time difference between current function call of MI_MD_Run and previous function call of MI_MD_Run.
MI_MD_GetYMean: Get the average value of the gray image.
MI_MD_GetLibVersion: Gets the version number of the library.
MI_MD_ComputeImageSAD: Gets the SAD statistics.

## 2.2. API Lists

### MI_MD_Init

Purpose

Initialize MD library.

Function Prototype

MD_HANDLE MI_MD_Init(U16 width, U16 height, U8 color, U8 w_div, U8 h_div);

Arguments

| Name | Description |
|---|---|
| width | Width of source image. |
| height | Height of source image. |
| color | channel of source image. |
| w_div | The number of divisions of window in horizontal direction. |
| h_div | The number of divisions of window in vertical direction. |

Return value

| Return value | Description |
|---|---|
| MD_HANDLE | Handle |

Requirement

Header files: mi_md.h

Library files: libMTE_LINUX.so

Note

- If use Y only for YUV image, please input 1 to "color" argument. (Currently only support Y only.)
- Enter low resolution to avoid high CPU usage, the recommended value is 320X180.

### MI_MD_Uninit

Purpose

Release memory of library.

Function Prototype

void MI_MD_Uninit(MD_HANDLE handle);

Arguments

| Name | Description |
|---|---|
| handle | Handle |

Return value

None.

Requirement

Header files: mi_md.h

Library files: libMTE_LINUX.so

Note

None.

## MI_MD_Run

Purpose

Run MD library.

Function Prototype

S32 MI_MD_Run(const U8* pImage);

Arguments

| Name | Description |
|---|---|
| pImage | Preview buffer address. |

Return value

| Return value | Description |
|---|---|
| -1 | Fail |
| other | total block numbers classified as motion. |

Requirement

Header files: mi_md.h

Library files: libMTE_LINUX.so

Note

- Run at low frame rate to avoid high CPU usage, the recommended value is 3~5.

## MI_MD_SetRegionInfo

Purpose

Set parameters for independent region motion detection.

Function Prototype

MI_RET MI_MD_SetRegionInfo(U8 num, MDBlockInfo_t* blk_info);

Arguments

| Name | Description |
|---|---|

| Name | Description |
|---|---|
| num | Number of independent regions. |
| blk_info | The structure of MDBlockInfo_t. |

Return value

| Return value | Description |
|---|---|
| MI_RET_SUCCESS | Success |
| MI_MD_RET_INVALID_PARAMETER | Parameters error |

Requirement

Header files: mi_md.h

Library files: libMTE_LINUX.sos

Note

- num: Number of regions, must be 2~64.
- It's allow regions overlap.
- Width of region being divisible by 8 is recommended for better performance.
- MI_MD_SetRegionInfo and MI_MD_SetDetectWindow can not use at the same time, can only choose one.

## MI_MD_SetDetectWindow

Purpose

Set motion detection window position.

Function Prototype

MI_RET MI_MD_SetDetectWindow(U16 lt_x, U16 lt_y, U16 rb_x, U16 rb_y, U8 w_div, U8 h_div);

Arguments

| Name | Description |
|---|---|
| lt_x | Horizontal position of window left-top point. |
| lt_y | Vertical position of window left-top point. |
| rb_x | Horizontal position of window right-bottom point. |
| rb_y | Vertical position of window right-bottom point. |
| w_div | The number of divisions of window in horizontal direction. |
| h_div | The number of divisions of window in vertical direction. |

Return value

| Return value | Description |
|---|---|

| Return value | Description |
|---|---|
| MI_RET_SUCCESS | Success |
| MI_MD_RET_INIT_ERROR | Initialization failed |
| MI_MD_RET_MALLOC_ERROR | Memory configuration error |
| MI_MD_RET_REGION_INIT_ERROR | Multiple region initialization failed |

Requirement

Header files: mi_md.h

Library files: libMTE_LINUX.so

Note

- w_div must smaller than 44, h_div must smaller than 36.
- Width of devided window being divisible by 8 is recommended for better performance.
- It is recommended that each detection window should not be less than 8 * 8 after segmentation, but it is not recommended to set too small.
- MI_MD_SetRegionInfo and MI_MD_SetDetectWindow can not use at the same time, can only choose one.

### MI_MD_GetDetectWindowSize

Purpose

Get motion detection window position.

Function Prototype

MI_RET MI_MD_GetDetectWindowSize(U16* st_x, U16* st_y, U16* div_w, U16* div_h);

Arguments

| Name | Description |
|---|---|
| st_x | Horizontal position of window left-top point. |
| st_y | Vertical position of window left-top point. |
| div_w | Horizontal pixels of each sub-window. |
| div_h | Vertical pixels of each sub-window. |

Return value

| Return value | Description |
|---|---|
| MI_RET_SUCCESS | Success |
| MI_MD_RET_INVALID_PARAMETER | Parameters error |
| MI_MD_RET_REGION_INIT_ERROR | Multiple region initialization failed |

Requirement

Header files: mi_md.h

Library files: libMTE_LINUX.so

Note

None.

### MI_MD_SetWindowParamsIn

Purpose

Set parameters for certain motion detection sub-window.

Function Prototype

MI_RET MI_MD_SetWindowParamsIn(U8 w_num, U8 h_num, const MDParamsIn_t* param);

Arguments

| Name | Description |
|---|---|
| w_num | Horizontal index of sub-window. |
| h_num | Vertical index of sub-window. |
| param | The structure of MDParamsIn_t. |

Return value

| Return value | Description |
|---|---|
| MI_RET_SUCCESS | Success |
| MI_MD_RET_INVALID_PARAMETER | Parameters error |
| MI_MD_RET_REGION_INIT_ERROR | Multiple region initialization failed |

Requirement

Header files: mi_md.h

Library files: libMTE_LINUX.so

Note

None.

### MI_MD_GetWindowParamsIn

Purpose

Get parameters for certain motion detection sub-window.

Function Prototype

MI_RET MI_MD_GetWindowParamsIn(U8 w_num, U8 h_num, MDParamsIn_t* param);

Arguments

| Name | Description |
|---|---|
| w_num | Horizontal index of sub-window. |
| h_num | Vertical index of sub-window. |
| param | The structure of MDParamsIn_t. |

Return value

| Return value | Description |
|---|---|
| MI_RET_SUCCESS | Success |
| MI_MD_RET_INVALID_PARAMETER | Parameters error |

Requirement

Header files: mi_md.h

Library files: libMTE_LINUX.so

Note

None.

## MI_MD_GetWindowParamsOut

Purpose

Get result for certain motion detection sub-window.

Function Prototype

MI_RET MI_MD_GetWindowParamsOut(U8 w_num, U8 h_num, MDParamsOut_t* param);

Arguments

| Name | Description |
|---|---|
| w_num | Horizontal index of sub-window. |
| h_num | Vertical index of sub-window. |
| param | The structure of MDParamsOut_t. |

Return value

| Return value | Description |
|---|---|
| MI_RET_SUCCESS | Success |
| MI_MD_RET_INVALID_PARAMETER | Parameters error |
| MI_MD_RET_REGION_INIT_ERROR | Multiple region initialization failed |

Requirement

Header files: mi_md.h

Library files: libMTE_LINUX.so

Note

None.

## MI_MD_SetTime

Purpose

Set the time difference between current function call of MI_MD_Run and previous function call of

MI_MD_Run.

Function Prototype

void MI_MD_SetTime(U32 time_diff);

Arguments

| Name | Description |
|---|---|
| time_diff | The time difference between current function call of MI_MD_Run and previous function call of MI_MD_Run. |

Return value

None.

Requirement

Header files: mi_md.h

Library files: libMTE_LINUX.so

Note

None.

## MI_MD_GetYMean

Purpose

Get the average value of the gray image.

Function Prototype

void MI_MD_GetYMean(U32 * mean);

Arguments

| Name | Description |
|---|---|
| mean | The average value of the gray image. |

Return value

None.

Requirement

Header files: mi_md.h

Library files: libMTE_LINUX.so

Note

None.

## MI_MD_GetLibVersion

Purpose

Gets the version number of the library.

Function Prototype

U32 MI_MD_GetLibVersion();

Arguments

None.

Return value

| Return value | Description |
|---|---|
| | Library version number. |

Requirement

Header files: mi_md.h

Library files: libMTE_LINUX.so

Note

None.

## MI_MD_ComputeImageSAD

Purpose

Gets the SAD(Sum of Absolute Difference) statistics.

Function Prototype

MI_RET MI_MD_ComputeImageSAD(const U8* pImage, void* psad_result, MDSAD_ctrl_t* psad_ctrl);

Arguments

| Name | Description |
|---|---|
| pImage | Preview buffer address. |
| psad_result | The memory address of the SAD result. |
| psad_ctrl | The structure of MDSAD_ctrl_t. |

Return value

| Return value | Description |
|---|---|
| MI_RET_SUCCESS | Success |
| MI_MD_RET_INVALID_HANDLE | Handle null. |
| MI_MD_RET_INVALID_PARAMETER | Parameters error |
| MI_MD_RET_MALLOC_ERROR | Memory configuration error |

Requirement

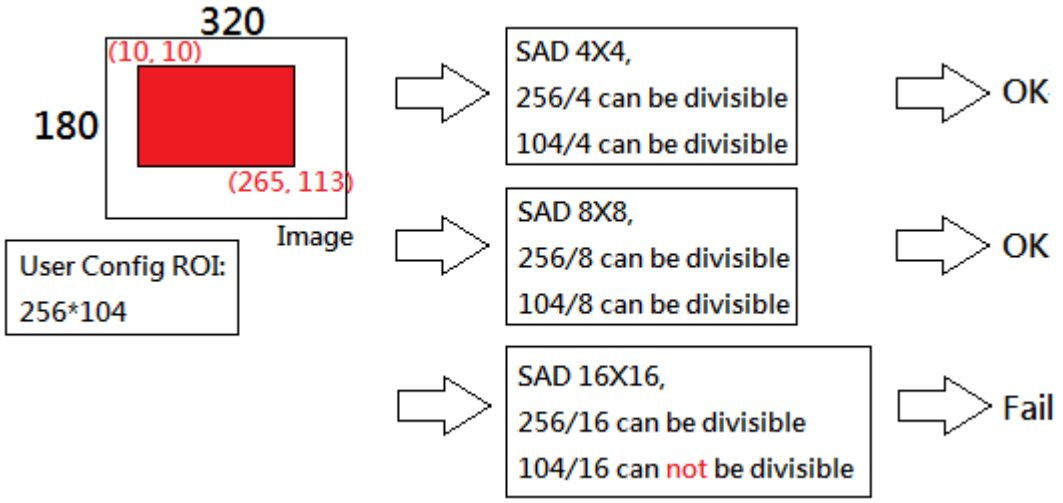
Header files: mi_md.h

Library files: libMTE_LINUX.so

Note

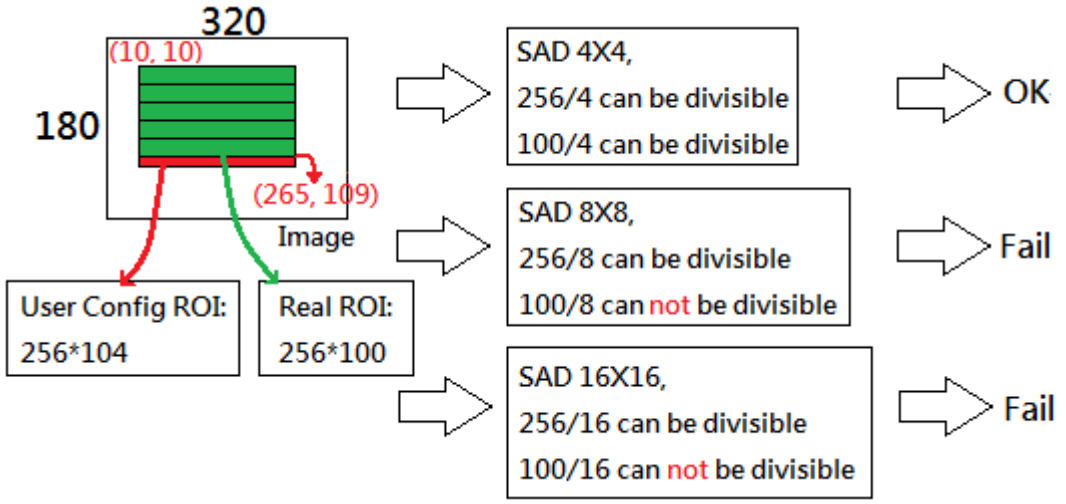
- Can not support multiple region mode. Its means you can not call MI_MD_SetRegionInfo before this, just MI_MD_SetDetectWindow.
- Alignment limit. Figure 1 shows ROI region not divided equally, call MI_MD_SetDetectWindow(10, 10, 265, 113, 1, 1) before SAD.

Figure 1




- Figure 2 shows ROI region be divided into 5 parts in vertical direction. Because we call MI_MD_SetDetectWindow(10, 10, 265, 113, 1, 5) , the extra 4 pixels will not be compute SAD.

Figure 2




- The buffer to save SAD result, take Figure 1 as an example,
  4x4 SAD, 8-bit output need allocated 256/4*104/4=1664 bytes,
  8x8 SAD, 8-bit output need allocated 256/8*104/8=416 bytes.
  Take Figure 2 as an example, 4x4 SAD, 8-bit output need allocated 256/4*100/4=1600 bytes.

# 3. DATA TYPE

## 3.1. Overview

| | |
|---|---|
| MDParamsIn_t☐ | Define the parameter of MD. |
| MDParamsOut_t☐ | Define the result of MD. |
| MDBlockInfo_t☐ | Define the range of independent region MD. |
| MDSAD_MODE_e | Defines SAD block size. |
| MDSAD_OUT_CTRL_e | Defines the SAD output control information. |
| MDSAD_ctrl_t | Defines SAD control parameters. |
| MI_RET | MD function return state. |

## 3.2. Struct Lists

### MDParamsIn_t

Description

Define the parameter of MD.

Syntax

```
typedef struct
{
  U8 enable;
  U8 size_perct_thd_min;
  U8 size_perct_thd_max;
  U8 sensitivity;
  U16 learn_rate;
} MDParamsIn_t;
```

Member

| Member | Description |
|---|---|
| enable | Is this window need to do MD. |
| size_perct_thd_min | The minimum percentage of pixels relative to sub-window total pixels are classified as foreground object. |
| size_perct_thd_max | The maximum percentage of pixels relative to sub-window total pixels are classified as foreground object. |
| sensitivity | It controls the threshold which determines if certain pixel would be |

| Member | Description |
|---|---|
| | classified as foreground object. If the value is small, pixels which have little difference with background are more easily to be classified as foreground object. |
| learn_rate | Units in millisecond. It controls how long the foreground object will be considered as background if it stops moving. |

Note

- size_perct_thd_min:0~99
- size_perct_thd_max:1~100, must be larger than size_perct_thd_min
- sensitivity: (10, 20, 30, ..., 100), 100 is the most sensitive. If the sensitivity is not multiple of 10, the feedback after the operation may not be the value of the original input, there will be +-1 deviation.
- learn_rate:1000~30000

## MDParamsOut_t

Description

Define the result of MD.

Syntax

```
typedef struct
{
 U8 md_result;
 U32 obj_cnt;
} MDParamsOut_t;
```

Member

| Member | Description |
|---|---|
| md_result | It stores the motion detection result for certain sub-window. If motion detection is true, then return 1, otherwise return 0. |
| obj_cnt | It stores total number of pixels classified as foreground object for certain sub-window. |

Note

None.

## MDBlockInfo_t

Description

Define the range of independent region MD.

Syntax

```
typedef struct
{
```

```
  U16 st_x;
  U16 st_y;
  U16 end_x;
  U16 end_y;
} MDBlockInfo_t;
```

Member

| Member | Description |
|---|---|
| st_x | Horizontal position of region left-top point. |
| st_y | Vertical position of region left-top point. |
| end_x | Horizontal position of region right-bottom point. |
| end_y | Vertical position of region right-bottom point. |

Note

None.

## MDSAD_MODE_e

Description

Defines SAD block size.

Syntax

```
typedef enum
{
  MDSAD_MODE_MB_4x4      = 0x0,
  MDSAD_MODE_MB_8x8      = 0x1,
  MDSAD_MODE_MB_16x16    = 0x2,
  MDSAD_MODE_BUTT
} MDSAD_MODE_e;
```

Member

| Member | Description |
|---|---|
| MDSAD_MODE_MB_4x4 | SAD block size is 4x4 |
| MDSAD_MODE_MB_8x8 | SAD block size is 8x8 |
| MDSAD_MODE_MB_16x16 | SAD block size is 16x16 |

Note

None.

## MDSAD_OUT_CTRL_e

Description

Defines the SAD output control information.

Syntax

```
typedef enum
{
  MDSAD_OUT_CTRL_16BIT_SAD      = 0x0,
  MDSAD_OUT_CTRL_8BIT_SAD       = 0x1,
  MDSAD_OUT_CTRL_BUTT
} MDSAD_OUT_CTRL_e;
```

Member

| Member | Description |
|---|---|
| MDSAD_OUT_CTRL_16BIT_SAD | Output 16-bit SAD value only |
| MDSAD_OUT_CTRL_8BIT_SAD | Output 8-bit SAD value only |

Note

None.

## MDSAD_ctrl_t

Description

Defines SAD control parameters.

Syntax

```
typedef struct
{
  MDSAD_MODE_e          enMode;
  MDSAD_OUT_CTRL_e      enOutCtrl;
} MDSAD_ctrl_t;
```

Member

| Member | Description |
|---|---|
| enMode | SAD operating mode.<br>For detail, see MDSAD_MODE_e |
| enOutCtrl | SAD output format.<br>For detail, see MDSAD_OUT_CTRL_e |

Note

None.

## MI_RET

Description

MD function return state.

Syntax

```
typedef enum _MI_RET_E
{
  MI_MD_RET_SUCCESS                 = 0x00000000,   /*MD API execution success*/
  MI_MD_RET_INIT_ERROR              = 0x10000401,   /*MD initialization error*/
  MI_MD_RET_IC_CHECK_ERROR          = 0x10000402,   /*Incorrect platform check for MD*/
  MI_MD_RET_INVALID_HANDLE          = 0x10000403,   /*Invalid MD handle*/
  MI_MD_RET_INVALID_PARAMETER       = 0x10000404,   /*Invalid MD parameter*/
  MI_MD_RET_MALLOC_ERROR            = 0x10000405,   /*Allocate MD working buffer error*/
  MI_MD_RET_REGION_INIT_ERROR       = 0x10000406,   /*Multi region initial error*/
} MI_MD_RET;
```

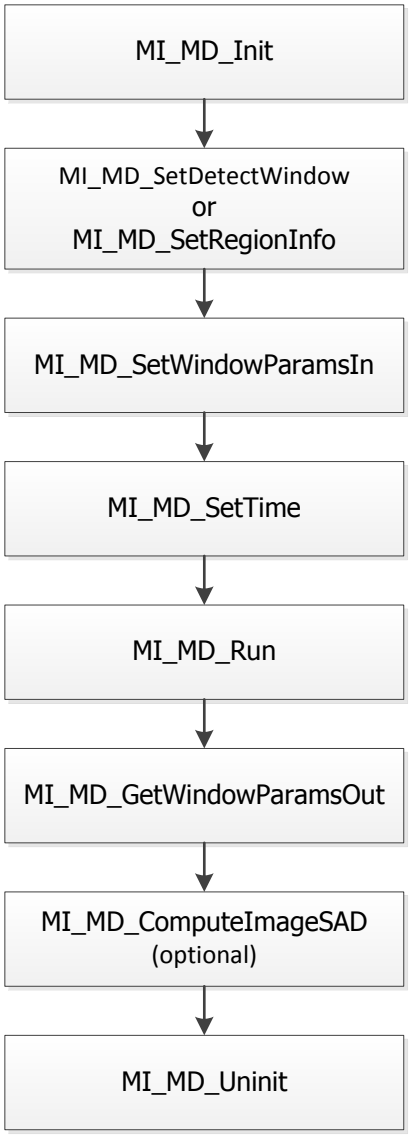
Member

| Member | Description |
|---|---|
| MI_RET_SUCCESS | Function success |
| MI_MD_RET_INIT_ERROR | MD initialize fail |
| MI_MD_RET_IC_CHECK_ERROR | IC check fail |
| MI_MD_RET_INVALID_HANDLE | MD handle is null |
| MI_MD_RET_INVALID_PARAMETER | Parameter error |
| MI_MD_RET_MALLOC_ERROR | Memory allocate error |
| MI_MD_RET_REGION_INIT_ERROR | Multi-region initialize fail |

Note

None.

# 4. MOTION DETECTION FLOW



# 5. EXAMPLE

Sample code: \IE\video\ MTE\I3\sample\MD\mi_sample_md.c