Advanced Series

High Performance JPEG Color Camera

Serial UART Interface

Protocol

June, 2018

SoooGuard INC

www.soooguard.com

Contents

Read Before 3

1. Standard Frame Format 3

2. Instruction of Command 4

2.1 Testing Protocol 4

2.1.1 Testing Camera (Cmd = 0x01) 4

2.2 Single Capture Protocol 4

2.2.1 Capture Image and Return Image Info. (Cmd = 0x40) 4

2.2.2 Request and Return Image (Cmd = 0x48, 0x49) 5

2.3 Continue Capture Protocol 5

2.3.1 Request Continue Capture (Cmd = 0X41, 0x42) 5

2.3.2 Request and Return Image File Name List (Cmd = 0X43, 0x44) 5

2.3.3 Request to Return Specified File Name Image Info. (Cmd = 0X45, 0x40) 6

2.4 OSD Protocol 6

2.4.1 OSD Control (Cmd = 0x52) 6

2.5 Fill-in Light Protocol 6

2.5.1 Fill-in Light Control (Cmd = 0x07) 6

2.6 Zoom Protocol 7

2.6.1 Reset Zoom (Cmd = 0x70, 0x71) 7

2.6.2 Zoom (Cmd = 0x72, 0x73) 7

2.6.3 Reset Focus (Cmd = 0x74, 0x75) 7

2.6.4 Reset Auto Refocus (Cmd = 0x76, 0x73) 7

2.7 Motion Detection Protocol 8

2.7.1 Set Motion Detection Area and Point (Cmd = 0x81, 0x82) 8

2.7.2 Time Synchronization (Cmd = 0x83, 0x84) 9

2.7.3 Motion Detection Alarm Information Notification (Cmd = 0x86) 9

2.8 Baud Rate and Address Protocol 9

2.8.1 Set Baud Rate and New Address (Cmd = 0x44, 0x45) 9

3. Appendix 10

3.1 CRC Checking Code 10

Read Before

1. This protocol applies to all Touiew Serial (RS485,RS232,TTL) JPEG Camera Series, including 0.3 Mega Pixels, 1.3 Mega Pixels, 2.0 Mega Pixels, 3.0 Mega Pixels and 5.0 Mega Pixels camera;
2. The image data can be transmitted in whole packet mode or sub-packet mode.
3. The image data will be stored in camera memory if there is no TF Card inserted.
4. While capturing at night, the fill-in light will turn on first and auto adjust the exposure time before capturing

1. Standard Frame Format

The frame consists of Header, Address, Command, Data Length, Data and CRC checksum. Total size is (8+D_4+D_5×256) bytes and Min. is 8 bytes.

D_0	**D_1**	**D_2**	**D_3**	**D_4**	**D_5**	**D_6 to D_n**	**C_1**	**C_2**
Header (2 bytes)		Address (1 byte)	Command (1 byte)	Data Length (2 bytes)		Data ((n-5) bytes)	CRC Checksum (2 bytes)	
H1	H2			LSB	MSB		LSB	MSB

Description:

(1) D_0, D_1: The frame header, fixed value: 0X90 0XEB;

(2) D_2: The camera Address (Add) or camera ID. Ranging from 0x01 to 0XFE. 0x00 and 0xFF is the reserved address(broadcast address). The camera will drop the protocol of any other address and not respond;

(3) D_3: means the command that camera can execute;

(4) D_4, D_5: Using little-endian. Represents the data length of D_6 to D_n. Ranges from 0~65535. Below calls LenL and LenM

(5) D_6 to D_n: Represents the real data.

(6) C_1, C_2: Using little-endian. Represents CRC checksum. Check from D_2(Address) to D_n. Total checking length=data length+4. White in testing mode, can use 0xC1, 0xC2 to instead. Below calls CRCL and CRCM

(7) Frame total size = Header + Address + Command + Data Length + Data + CRC Checksum

= 2 + 1 + 1 + 2 + (n-5) + 2

= 8 + D_4 + D_5×256

2. Instruction of Command

2.1 Testing Protocol

2.1.1 Testing Camera (Cmd = 0x01)

Function: Uses this command to test camera working status.

Data Bit	D_0	D_1	D_2	D_3	D_4	D_5	D_6 to D_n	C_1	C_2
Name	H1	H2	Add	Cmd	LenL	LenM	Data	CRCL	CRCM
Send	**90**	**EB**	**XX**	**01**	**02**	**00**	**55 AA**	**C1**	**C2**
Bytes	1	1	1	1	1	1	2	1	1
Return	**90**	**EB**	**XX**	**01**	**03**	**00**	**00 AA 55**	**F6**	**EB**
Bytes	1	1	1	1	1	1	3	1	1

Example:

Action	Hexadecimal Notation	Description
Send	90 EB 01 01 02 00 55 AA C1 C2	Host Device send testing command to confirm the camera status
Return	90 EB 01 01 03 00 00 AA 55 F6 EB	The camera is in good working condition

2.2 Single Capture Protocol

2.2.1 Capture Image and Return Image Info. (Cmd = 0x40)

Function: Camera will capture an image, divide into number of sub-packets and store to memory.

Data Bit	D_0	D_1	D_2	D_3	D_4	D_5	D_6 to D_n						C_1	C_2
Name	H1	H2	Add	Cmd	LenL	LenM	Data						CRCL	CRCM
Send	**90**	**EB**	**XX**	**40**	**04**	**00**	**Sub-packet Size**		**Resolution**		**Compression**		**C1**	**C2**
Bytes	1	1	1	1	1	1	2		1		1		1	1
Return	**90**	**EB**	**XX**	**40**	**0B**	**00**	**Status**	**Image Size**	**Sub-packets No.**	**Sub-packet Size**	**Resolution**	**Compression**	**XX**	**XX**
Bytes	1	1	1	1	1	1	1	4	2	2	1	1	1	1

★ **Sub-packet Size:** can set to be 32B, 64B, 256B, 512B, 768B, 1024B, 2048B, 4096B and Max 65K**; Can be ignored.**

★ **Resolution Instruction**

Command	Resolution	Aspect Ratio	Remark	Command	Resolution	Aspect Ratio	Remark
0x01	160*120	4:3	QQVGA	0x20	3648*2736	4:3	10MP
0x03	320*240	4:3	QVGA	0x21	4000*3000	4:3	12MP
0x05	640*480	4:3	0.3MP	0x30	480*270	16:9	0.1MP
0x06	1280*960	4:3	1.3MP	0x31	640*360	16:9	0.2MP
0x07	800*600	4:3	0.5MP	0x32	800*450	16:9	0.4MP
0x08	1024*768	4:3	0.8MP	0x33	960*540	16:9	0.5MP
0x09	1280*800	4:3	1.0MP	0x34	1024*576	16:9	0.6MP
0x10	1600*1024	25:16	Over 2.0MP supported	~~0x35~~	~~1280*720~~	~~16:9~~	~~Standard 1.0MP~~
0x11	1600*1200	4:3	Over 2.0MP supported	0x36	1366*768	16:9	1.0MP
0x12	2048*1536	4:3	Standard 3.0MP	0x37	1440*810	16:9	1.2MP
0x13	2560*1920	4:3	Standard 5.0MP	0x38	1600*900	16:9	1.4MP
0x14	2592*1944	4:3	Over 5.0MP supported	~~0x39~~	~~1920*1080~~	~~16:9~~	~~Standard 2.0MP~~
0x15	1280*720	16:9	Standard 1.0MP	0x40	2560*1440	16:9	3.7MP
0x16	1920*1080	16:9	Standard 2.0MP	0x41	3200*1800	16:9	5.0MP
0x17	1280*1024	5:4	Standard 1.3MP	0x42	3840*2160	16:9	8.0MP
0x18	3000*2000	3:2	6.0MP	If beyond max resolution, the camera will execute the max			
0x19	3264*2448	4:3	8.0MP				

★ **Compression:** Range from 1 to 5, smaller value means compress less and better image quality;

★ **Status:** Default is 0x00;

★ **Image Size:** The size of the image;

★ **Sub-packets No.:** Total No. of the image packets**. But while get image data, you can get the any size data by your own.**

Example:

Action	Hexadecimal Notation	Description
Send	90 EB 01 40 04 00 00 02 05 01 C1 C2	Ask NO. 1 camera to capture a 640×480 image that divided into 512B each sub-packet with compression level 1
Return	90 EB 01 40 0B 00 00 83 16 00 00 0B 00 00 02 05 01 27 B5	Camera take a 5KB image of 11 sub-packets

Remark: Camera will capture image with default resolution and compression rate if their parameters are invalid

2.2.2 Request and Return Image (Cmd = 0x48, 0x49)

Function: Request camera to send image data from specified start of address with specified length (sub-packet size)

Data Bit	**D_0**	**D_1**	**D_2**	**D_3**	**D_4**	**D_5**	**D_6 to D_n**		**C_1**	**C_2**
Name	H1	H2	Add	Cmd	LenL	LenM	Data		CRCL	CRCM
Send	**90**	**EB**	**XX**	**48**	**06**	**00**	**Start of Address**	**Specified Image Data Length**	**C1**	**C2**
Bytes	1	1	1	1	1	1	4	2	1	1
Send	**90**	**EB**	**XX**	**49**	**Specified Image Data Length**		**Specified Image Data**		**C1**	**C2**
Bytes	1	1	1	1	1	1	**Specified Image Data Length**		1	1

★ **Start of Address:** The start address of the image data that want to get. It begins with 0x00 00 00 00;

★ **Specified Image Data Length:** Same as sub-packets size above. Max 65K**.**

Example:

Action	**Hexadecimal Notation**	**Description**
Send	90 EB 01 48 06 00 00 02 00 00 00 02 C1 C2	Ask NO. 1 camera to send 512B image data from the start address of 0x00 02 00 00
Return	90 EB 01 49 00 03 D_0 D_1 … D_{n-1} D_n C1 C2	NO. 1 camera return 768B image data: D_0 D_1 … D_{n-1} D_n

Remark: The camera will not return image data if the obtaining sub-packet length beyond packet stored in camera memory.

2.3 Continue Capture Protocol

2.3.1 Request Continue Capture (Cmd = 0X41, 0x42)

Function: Request camera to capture images constantly with specified image quantity and interval

Data Bit	**D_0**	**D_1**	**D_2**	**D_3**	**D_4**	**D_5**	**D_6 to D_n**										**C_1**	**C_2**
Name	H1	H2	Add	Cmd	LenL	LenM	Data										CRCL	CRCM
Send	**90**	**EB**	**01**	**41**	**0A**	**00**	**Resolution**	**Compression**	**Qty**	**Interval**	**Y**	**M**	**D**	**H**	**M**	**S**	**C1**	**C2**
Bytes	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Return1	**90**	**EB**	**XX**	**42**	**02**	**00**	**AA**			**00**							**C1**	**C2**
Return2	**90**	**EB**	**XX**	**42**	**02**	**00**	**AA**			**08**							**C1**	**C2**
Bytes	1	1	1	1	1	1	1			1							1	1

★ **Qty:** Image quantity that ask the camera to take;

★ **Interval:** Time interval for capturing a new image**;**

★ **Y:** Year=This Year-2000;

★ **M:** Month;

★ **D:** Day;

★ **H:** Hour;

★ **M:** Minute;

★ **S:** Second;

★ **File name format:** File name format is “YYMMDD HHMMSS.jpg”. Totally 18 bytes.

Example:

Action	**Hexadecimal Notation**	**Description**
Send	90 EB 01 41 0A 00 05 01 05 00 12 05 08 10 11 12 C1 C2	Ask No. 1 camera for continue capture 5 compression level 1 0.3mp images at 0s interval begin named after the time of 2018-5-8 10:11:12
Return1	90 EB 01 42 02 00 AA 00 96 0F	Continue capture start
Return2	90 EB 01 42 02 00 AA 08 9E 8E	Continue capture completed

2.3.2 Request and Return Image File Name List (Cmd = 0X43, 0x44)

Function: Request camera to send images file name list that stored in camera

Data Bit	**D_0**	**D_1**	**D_2**	**D_3**	**D_4**	**D_5**	**D_6 to D_n**										**C_1**	**C_2**
Name	H1	H2	Add	Cmd	LenL	LenM	Data										CRCL	CRCM
Send	**90**	**EB**	**01**	**43**	**0A**	**00**	**Sequence No.**	**Qty**	**Time Direction**	**Reserve**	**Y**	**M**	**D**	**H**	**M**	**S**	**C1**	**C2**
Bytes	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Return1	**90**	**EB**	**XX**	**44**	**XX**	**XX**	**Image File Name List**										**C1**	**C2**
Bytes	1	1	1	1	1	1	18 bytes * Qty										1	1

★ **Sequence No.:** The serial No. of picture for the help of selection. For example, “0x08” means sending the image files name after the 8th picture from the selected time pont; **Set 0x00 from the selected time.**

★ **Qty:** Image file name quantity that request camera to send;

★ **Time Direction:** Before or after the specified time. “**0x00**” mean before the specified time while “**0x01**” mean after the specified time**;**

★ **Reserve:** 0xBB;
★ **Y:** Year=This Year-2000. For example, “0x12” means “2018”;
★ **M:** Month;
★ **D:** Day;
★ **H:** Hour;
★ **M:** Minute;
★ **S:** Second;

Example:

Action	Hexadecimal Notation	Description
Send	90 EB 01 43 0A 00 08 66 01 BB 11 01 02 03 04 05 C1 C2	Ask No. 1 camera to send the file names before 2017-1-2 3:4:5 after the 8th picture

Remark: if time parameters are all “0x0”, that means current time, only can search before.

2.3.3 Request to Return Specified File Name Image Info. (Cmd = 0X45, 0x40)

Function: Request camera to return specified images file info.

Data Bit	D_0	D_1	D_2	D_3	D_4	D_5	D_6 to D_n						C_1	C_2
Name	H1	H2	Add	Cmd	LenL	LenM	Data						CRCL	CRCM
Send	**90**	**EB**	**01**	**45**	**14**	**00**	**Sub-packet Size**		**Image File Name**				**C1**	**C2**
Bytes	1	1	1	1	1	1	2		18				1	1
Return1	**90**	**EB**	**XX**	**40**	**0B**	**00**	**Status**	**Image Size**	**Sub-packets No.**	**Sub-packet Size**	**Resolution**	**Compression**	**C1**	**C2**
Bytes	1	1	1	1	1	1	1	4	2	2	1	1	1	1

Example:

Action	Hexadecimal Notation	Description
Send	90 EB 01 45 14 00 00 02 31 38 30 33 30 34 30 35 30 36 30 37 2E 6A 70 67 00 C1 C2	Request No.1 Camera to send the image of file name: 180304 050607.jpg and divided into size of 512B
Return	90 EB 01 40 0B 00 00 83 16 00 00 0B 00 00 02 FF FF C1 C2	Camera dived the 5KB image into 11 sub-packets

Remark: Use 0XFF 0xFF to into the resolution and compression as they make no sense in this mode.

2.4 OSD Protocol

2.4.1 OSD Control (Cmd = 0x52)

Function: Display Character on the image

Data Bit	D_0	D_1	D_2	D_3	D_4	D_5	D_6 to D_n				C_1	C_2
Name	H1	H2	Add	Cmd	LenL	LenM	Data				CRCL	CRCM
Send	**90**	**EB**	**XX**	**52**	**XX**	**XX**	**X-Position**	**Y-Position**	**Font Height**	**Character**	**C1**	**C2**
Bytes	1	1	1	1	1	1	2	2	1	n-11	1	1

Example:

Action	Hexadecimal Notation	Description
Send	90 EB 01 52 0A 00 06 00 08 00 0C 31 32 33 34 65 C1 C2	Display character of “1234e” at the point of (6,8) from up left corner with font height of 12 pixels

Remark: The character is according to ASCII code. The font height is invalid. Set 0x00 and depended by the system.

2.5 Fill-in Light Protocol

2.5.1 Fill-in Light Control (Cmd = 0x07)

Function: Control the fill-in light mode to be manual or auto. Open or close the fill-in light

Data Bit	D_0	D_1	D_2	D_3	D_4	D_5	D_6 to D_n		C_1	C_2
Name	H1	H2	Add	Cmd	LenL	LenM	Data		CRCL	CRCM
Send	**90**	**EB**	**XX**	**07**	**02**	**00**	**Status**	**Mode**	**C1**	**C2**
Bytes	1	1	1	1	1	1	1	1	1	1

★ **Status:** “0x33” means open fill-in light and “0xCC” means close fill-in light;
★ **Mode:** “0x00” means manual mode and “0x01” means auto mode.

Example:

Action	Hexadecimal Notation	Description
Send	90 EB 01 07 02 00 CC 00 C1 C2	Manual Mode, Fill-in light closed

Send	90 EB 01 07 02 00 33 00 C1 C2	Manual Mode, Fill-in light opened
Send	90 EB 01 07 02 00 33 01 C1 C2	Auto Mode

2.6 Zoom Protocol

2.6.1 Reset Zoom (Cmd = 0x70, 0x71)

Function: Reset camera focal distance to the short focal length

Data Bit	D_0	D_1	D_2	D_3	D_4	D_5	D_6 to D_n		C_1	C_2
Name	H1	H2	Add	Cmd	LenL	LenM	Data		CRCL	CRCM
Send	**90**	**EB**	**XX**	**70**	**02**	**00**	**70**	**70**	**C1**	**C2**
Bytes	1	1	1	1	1	1	1	1	1	1
Return	**90**	**EB**	**XX**	**71**	**02**	**00**	**71**	**71**	**C1**	**C2**
Bytes	1	1	1	1	1	1	1	1	1	1

Remark: This command are used in the condition of suddenly loss power or not bad zoom effect

2.6.2 Zoom (Cmd = 0x72, 0x73)

Function: Zoom

Data Bit	D_0	D_1	D_2	D_3	D_4	D_5	D_6 to D_n				C_1	C_2
Name	H1	H2	Add	Cmd	LenL	LenM	Data				CRCL	CRCM
Send	**90**	**EB**	**XX**	**72**	**02**	**00**	**YY**		**YY**		**C1**	**C2**
Bytes	1	1	1	1	1	1	1		1		1	1
Return	**90**	**EB**	**XX**	**73**	**02**	**00**	**YY**	**YY**	**ZZ**	**ZZ**	**C1**	**C2**
Bytes	1	1	1	1	1	1	1	1	1	1	1	1

★ **YY:** Represents the current focal length location of the lens total focus length. Range from 0-100. 0 means short focal length. 100 means long focal length. 50 means the middle. If the two YY YY value are larger than 100, the zoom will not work;

★ **ZZ:** Represents focusing location. Camera will be auto focus;

Example:

Action	Hexadecimal Notation	Description
Send	90 EB 01 72 02 00 32 32 C1 C2	Set camera to be the location of 32% total focal length
Return	90 EB 01 73 04 00 32 32 25 25 C1 C2	Camera zoom and auto focus successfully

Remark: The camera will auto focus after zooming. The auto focus time will take about several seconds to more than 10 seconds which is depended by the outside light condition.

2.6.3 Reset Focus (Cmd = 0x74, 0x75)

Function: Reset camera focusing

Data Bit	D_0	D_1	D_2	D_3	D_4	D_5	D_6 to D_n		C_1	C_2
Name	H1	H2	Add	Cmd	LenL	LenM	Data		CRCL	CRCM
Send	**90**	**EB**	**XX**	**74**	**02**	**00**	**74**	**74**	**C1**	**C2**
Bytes	1	1	1	1	1	1	1	1	1	1
Return	**90**	**EB**	**XX**	**75**	**02**	**00**	**75**	**75**	**C1**	**C2**
Bytes	1	1	1	1	1	1	1	1	1	1

Remark: This command are used in the condition of suddenly loss power or not bad zoom effect

2.6.4 Reset Auto Refocus (Cmd = 0x76, 0x73)

Function: After zooming, the camera will execute this command to reset refocus automatically

Data Bit	D_0	D_1	D_2	D_3	D_4	D_5	D_6 to D_n				C_1	C_2
Name	H1	H2	Add	Cmd	LenL	LenM	Data				CRCL	CRCM
Send	**90**	**EB**	**XX**	**76**	**02**	**00**	**YY**		**YY**		**C1**	**C2**
Bytes	1	1	1	1	1	1	1		1		1	1
Return	**90**	**EB**	**XX**	**73**	**02**	**00**	**YY**	**YY**	**ZZ**	**ZZ**	**C1**	**C2**
Bytes	1	1	1	1	1	1	1	1	1	1	1	1

2.7 Motion Detection Protocol

2.7.1 Set Motion Detection Area and Point (Cmd = 0x81, 0x82)

Function: Camera has divided the image into 108(12×9) areas. Then set detective points for each area. While moving in the selected detective area, the camera will continue capture picture and send alarm information to host

Data Bit	D_0	D_1	D_2	D_3	D_4	D_5	D_6 to D_n					C_1	C_2
Name	H1	H2	Add	Cmd	LenL	LenM	Data					CRCL	CRCM
Send	**90**	**EB**	**XX**	**81**	**70**	**00**	**Sensitivity**	**Qty.**	**Density**	**Reserve**	**Area**	**C1**	**C2**
Bytes	1	1	1	1	1	1	1	1	1	1	108	1	1
Return	**90**	**EB**	**XX**	**82**	**00**	**00**						**C1**	**C2**
Bytes	1	1	1	1	1	1						1	1

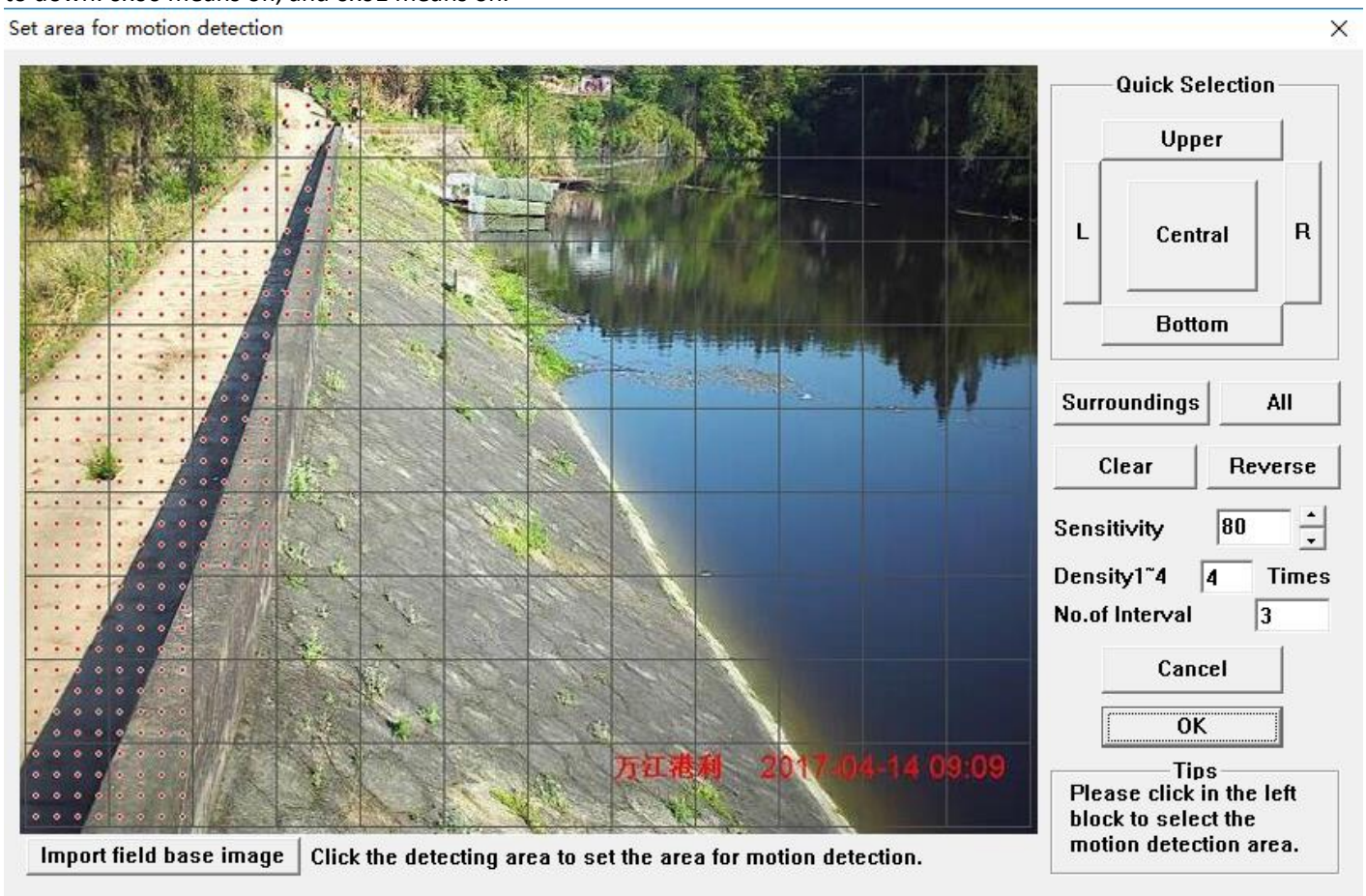
★ **Sensitivity:** Range from 0-100. Larger value easier to trigger the alarm and recommended value is “80”. If set “0” , camera will disarm and larger “0” the camera will arm;

★ **Qty:** Represents the quantity of image while camera is triggered alarm;

★ **Density:** Range from 1-4. 1 represents 16 points and 4 represents 256 points in each area. Larger value, more detective points in unit area and more suitable to detect the smaller object. While smaller value, less detective points in unit area and more suitable to in close detection;

★ **Reserve:** Use 0X00

★ **Area:** Total area no. is 12×9=108. Need to put 108 bytes to show on-off status of the detected area from left to right, up to down. 0x00 means off, and 0x01 means on.



Example:

Action	Hexadecimal Notation	Description
Send	90 EB 01 81 70 00 50 05 04 00 01 01 01 01 01 01 01 01 01 00 00 … 00 00 00 00 00 00 00 C1 C2	Request No. 1 camera to set 1st line of the all areas to be detected with 80 sensitivity and level-4 density. Take 5 images while alarm
Send	90 EB 01 82 00 00 C1 C2	Set alarm area and point successfully

2.7.2 Time Synchronization (Cmd = 0x83, 0x84)

Function: Sync camera time. After losing power, camera will reset and the motion detection information is named after the time. So if no time Synchronization, the alarm time can not be determined.

Data Bit	D_0	D_1	D_2	D_3	D_4	D_5	D_6 to D_n						C_1	C_2
Name	H1	H2	Add	Cmd	LenL	LenM	Data						CRCL	CRCM
Send	**90**	**EB**	**XX**	**83**	**06**	**00**	**Year**	**Month**	**Day**	**Hour**	**Minute**	**Second**	**C1**	**C2**
Bytes	1	1	1	1	1	1	1	1	1	1	1	1	1	1
Return	**90**	**EB**	**XX**	**82**	**00**	**00**							**C1**	**C2**
Bytes	1	1	1	1	1	1							1	1

★ **Year:** Current time- 2000;

Example:

Action	Hexadecimal Notation	Description
Send	90 EB 01 83 06 00 12 01 02 03 04 05 C1 C2	Set camera time to be 2018-1-2 3:4:5
Send	90 EB 01 84 00 00 C1 C2	Set time successfully

2.7.3 Motion Detection Alarm Information Notification (Cmd = 0x86)

Function: Camera send back the alarm notification (time) to host. Default this function is turn off.

Data Bit	D_0	D_1	D_2	D_3	D_4	D_5	D_6 to D_n						C_1	C_2
Name	H1	H2	Add	Cmd	LenL	LenM	Data						CRCL	CRCM
Send	**90**	**EB**	**XX**	**86**	**06**	**00**	**Year**	**Month**	**Day**	**Hour**	**Minute**	**Second**	**C1**	**C2**
Bytes	1	1	1	1	1	1	1	1	1	1	1	1	1	1

★ **Year:** Current time- 2000;

Example:

Action	Hexadecimal Notation	Description
Return	90 EB 01 86 06 00 12 01 02 03 04 05 C1 C2	A motion detection alarm happened at 2018-1-2 3:4:5

2.8 Baud Rate and Address Protocol

2.8.1 Set Baud Rate and New Address (Cmd = 0x44, 0x45)

Function: Change the camera baud rate and address

Data Bit	D_0	D_1	D_2	D_3	D_4	D_5	D_6 to D_n				C_1	C_2
Name	H1	H2	Add	Cmd	LenL	LenM	Data				CRCL	CRCM
Send	**90**	**EB**	**XX**	**44**	**04**	**00**	**Baud Rate**	**Saving Mode**	**New Add**	**New Add**	**C1**	**C2**
Bytes	1	1	1	1	1	1	1	1	1	1	1	1
Return	**90**	**EB**	**XX**	**45**	**04**	**00**	**Baud Rate**	**Saving Status**	**New Add**	**New Add**	**C1**	**C2**
Bytes	1	1	1	1	1	1	1	1	1	1	1	1

★ **Baud Rate**

Command	Baud Rate
0x00	No changing
0x01	9600bps
0x02	19200bps
0x04	38400bps
0x05	57600bps
0x06	115200bps

★ **Saving Mode:** 0x01 means new baud rate will be saved to camera also after losing power while 0x00 will not;

★ **New Add:** No changing for address if new address is set to be “0x00” or “0xFF” as they are broadcast address;

★ **Saving Status:** 0x01 means new baud rate is saved to camera also after losing power;

Example:

Action	Hexadecimal Notation	Description
Send	90 EB 01 44 04 00 02 01 0A 0A C1 C2	Set camera to new baud rate of 19200bps and new address 10 to No. 1 camera
Send	90 EB 01 45 04 00 02 01 0A 0A C1 C2	Set new baud rate and new address successfully

3. Appendix

3.1 CRC Checking Code

//Crc16 Calculating Function

```
const unsigned short    crc_ta[256]={ /* CRC Remains Lookup Table */

0x0000, 0x1021, 0x2042, 0x3063, 0x4084, 0x50a5, 0x60c6, 0x70e7,
0x8108, 0x9129, 0xa14a, 0xb16b, 0xc18c, 0xd1ad, 0xe1ce, 0xf1ef,
0x1231, 0x0210, 0x3273, 0x2252, 0x52b5, 0x4294, 0x72f7, 0x62d6,
0x9339, 0x8318, 0xb37b, 0xa35a, 0xd3bd, 0xc39c, 0xf3ff, 0xe3de,
0x2462, 0x3443, 0x0420, 0x1401, 0x64e6, 0x74c7, 0x44a4, 0x5485,
0xa56a, 0xb54b, 0x8528, 0x9509, 0xe5ee, 0xf5cf, 0xc5ac, 0xd58d,
0x3653, 0x2672, 0x1611, 0x0630, 0x76d7, 0x66f6, 0x5695, 0x46b4,
0xb75b, 0xa77a, 0x9719, 0x8738, 0xf7df, 0xe7fe, 0xd79d, 0xc7bc,
0x48c4, 0x58e5, 0x6886, 0x78a7, 0x0840, 0x1861, 0x2802, 0x3823,
0xc9cc, 0xd9ed, 0xe98e, 0xf9af, 0x8948, 0x9969, 0xa90a, 0xb92b,
0x5af5, 0x4ad4, 0x7ab7, 0x6a96, 0x1a71, 0x0a50, 0x3a33, 0x2a12,
0xdbfd, 0xcbdc, 0xfbbf, 0xeb9e, 0x9b79, 0x8b58, 0xbb3b, 0xab1a,
0x6ca6, 0x7c87, 0x4ce4, 0x5cc5, 0x2c22, 0x3c03, 0x0c60, 0x1c41,
0xedae, 0xfd8f, 0xcdec, 0xddcd, 0xad2a, 0xbd0b, 0x8d68, 0x9d49,
0x7e97, 0x6eb6, 0x5ed5, 0x4ef4, 0x3e13, 0x2e32, 0x1e51, 0x0e70,
0xff9f, 0xefbe, 0xdfdd, 0xcffc, 0xbf1b, 0xaf3a, 0x9f59, 0x8f78,
0x9188, 0x81a9, 0xb1ca, 0xa1eb, 0xd10c, 0xc12d, 0xf14e, 0xe16f,
0x1080, 0x00a1, 0x30c2, 0x20e3, 0x5004, 0x4025, 0x7046, 0x6067,
0x83b9, 0x9398, 0xa3fb, 0xb3da, 0xc33d, 0xd31c, 0xe37f, 0xf35e,
0x02b1, 0x1290, 0x22f3, 0x32d2, 0x4235, 0x5214, 0x6277, 0x7256,
0xb5ea, 0xa5cb, 0x95a8, 0x8589, 0xf56e, 0xe54f, 0xd52c, 0xc50d,
0x34e2, 0x24c3, 0x14a0, 0x0481, 0x7466, 0x6447, 0x5424, 0x4405,
0xa7db, 0xb7fa, 0x8799, 0x97b8, 0xe75f, 0xf77e, 0xc71d, 0xd73c,
0x26d3, 0x36f2, 0x0691, 0x16b0, 0x6657, 0x7676, 0x4615, 0x5634,
0xd94c, 0xc96d, 0xf90e, 0xe92f, 0x99c8, 0x89e9, 0xb98a, 0xa9ab,
0x5844, 0x4865, 0x7806, 0x6827, 0x18c0, 0x08e1, 0x3882, 0x28a3,
0xcb7d, 0xdb5c, 0xeb3f, 0xfb1e, 0x8bf9, 0x9bd8, 0xabbb, 0xbb9a,
0x4a75, 0x5a54, 0x6a37, 0x7a16, 0x0af1, 0x1ad0, 0x2ab3, 0x3a92,
0xfd2e, 0xed0f, 0xdd6c, 0xcd4d, 0xbdaa, 0xad8b, 0x9de8, 0x8dc9,
0x7c26, 0x6c07, 0x5c64, 0x4c45, 0x3ca2, 0x2c83, 0x1ce0, 0x0cc1,
0xef1f, 0xff3e, 0xcf5d, 0xdf7c, 0xaf9b, 0xbfba, 0x8fd9, 0x9ff8,
0x6e17, 0x7e36, 0x4e55, 0x5e74, 0x2e93, 0x3eb2, 0x0ed1, 0x1ef0
};
```

//CRC Check: This is used to checking the continuous data;

```
unsigned short Crc16(unsigned char *ptr, unsigned short len)
{
    unsigned short crc;
    unsigned char da;
```

```
    crc=0;
    while(len--!=0)
      {
        da=(unsigned char) (crc/256);    // Store the CRC high-8bit temporary with 8-bitbinary number format
        crc<<=8; //Left shit 8 bit；
      crc^=crc_ta[da^*ptr]; // high 8-bit plus current CRC then calculate the new CRC by looking up from the CRC Remains Lookup Table, then
 plus the previous CRC value
        ptr++;
    }
    return(crc);
}
```

//CRC Check: This is used to check the discontiguous data;

```
unsigned short    Crc16_New(unsigned short old_crc,unsigned char *ptr, unsigned short len)
{
    unsigned short crc;
    unsigned char    da;

    crc=old_crc;
    while(len--!=0)
      {
        da=(unsigned char) (crc/256); // Store the CRC high-8bit temporary with 8-bitbinary number format
        crc<<=8; //Left shit 8 bit；
      crc^=crc_ta[da^*ptr]; / high 8-bit plus current CRC then calculate the new CRC by looking up from the CRC Remains Lookup Table, then
 plus the previous CRC value
        ptr++;
    }
    return(crc);
}
```