EasyLensMotor

A COM Component to access lens motors include focus and Zoom motor

Juiwen Hsu@2013-0530

Logitech Confidential - For Internal Use Only

Logitech Inc.

1 Content

Content

2 Introduction

2.1 Purpose

EasyLensMotor COM Component provide a simple way to access lens motor via Logitech USB Video Class extension unit.

2.2 Revision History

Revision JuiwenHsu@2013-0530: Initial draft version

2.3 Acknowledgments

Contributions to this document were made by the following people:
Name, Company

3 Setup

Before start with EasyLensMotor, you should install below components in you system:

(1) EasyAVEngine version 2013.05.30.XX or latter

3.1 EasyAVEngine

This Apply to EasyAVEngine version 2013.05.30.XX or latter, the file EasyAVRuntimeEngineX86-Ver(2013.05.30.XX).ZIP or latter.

Download from [Google Drive]: https://drive.google.com/a/logitech.com/?tab=mo#folders/0B60SgTBmsbI8NkZPb0hKSThrR3M
or

Download from: \\us01f01\video\video\Hardware Engineering\TDE Collections\EasyAVEngine\EndUsers\EasyAVRuntimeEngine\

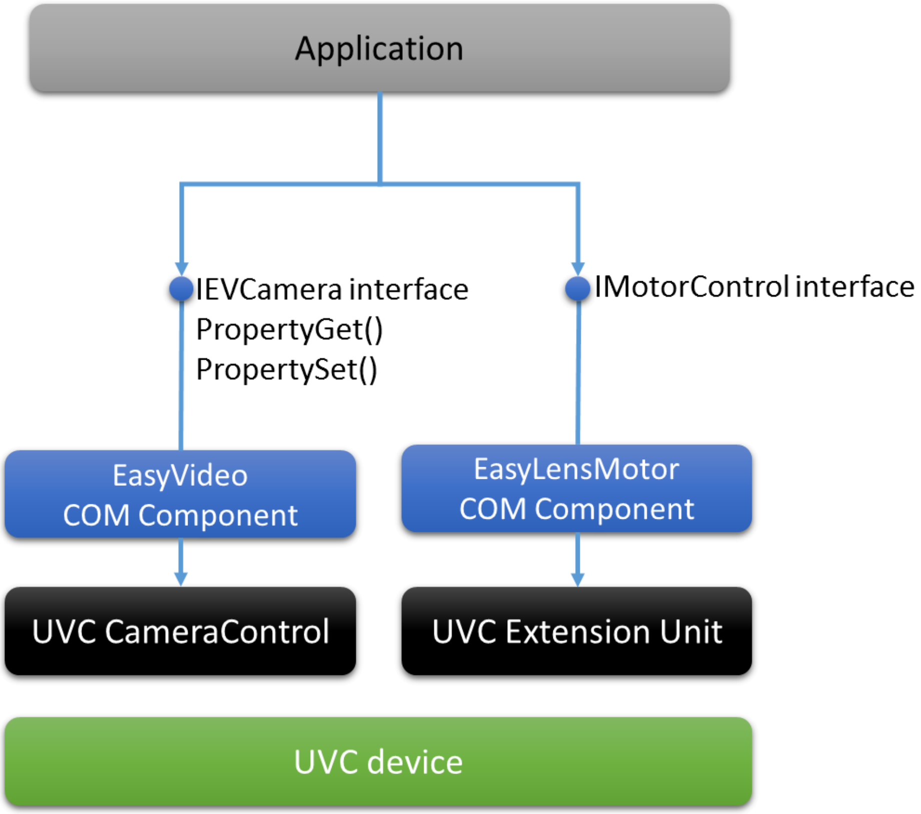
The file EasyAVRuntimeEngineX86.exe under that download folder is always the newest one.

4 EasyLensMotor COM Component

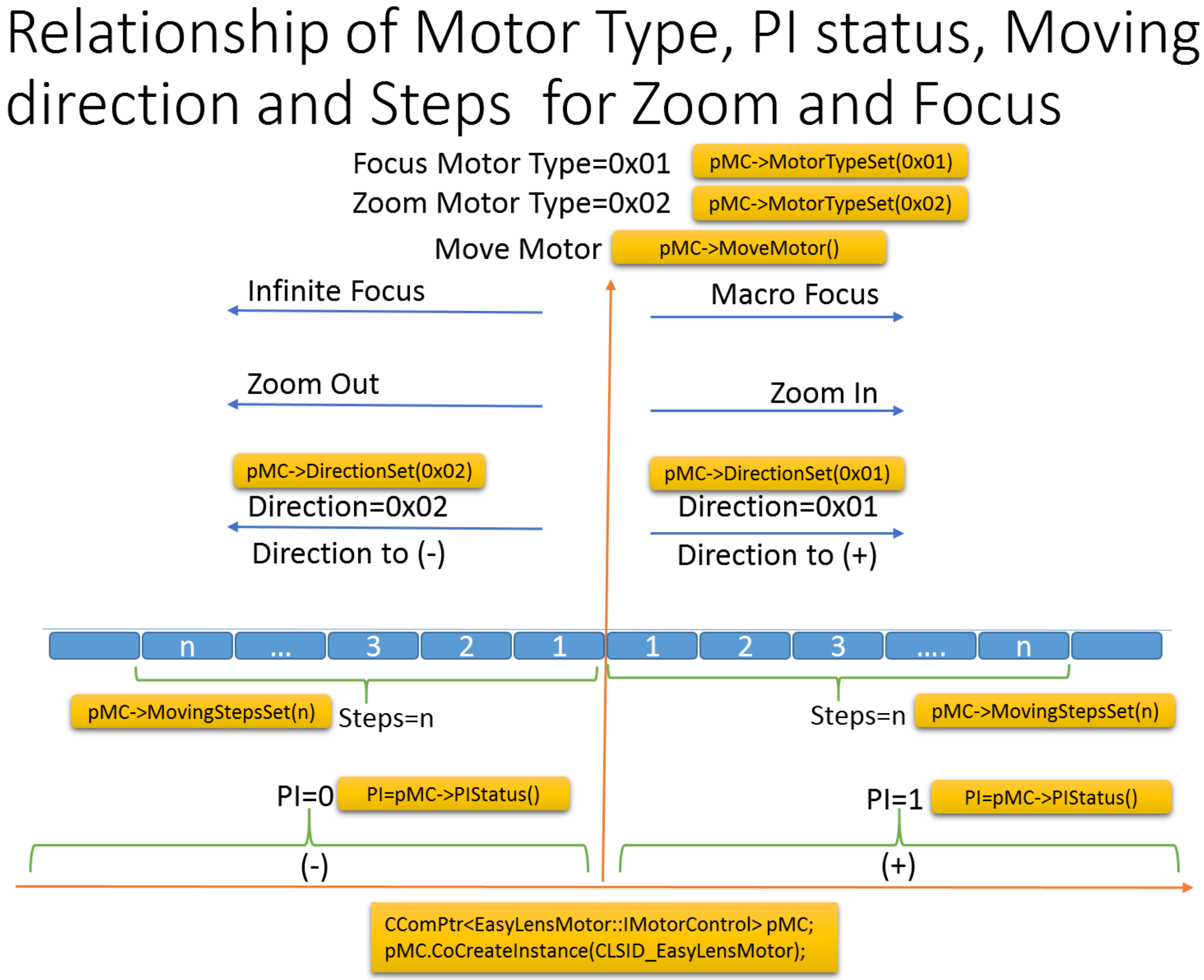
EasyLensMotor COM Component provide a simple way to access lens motor via Logitech USB Video Class extension unit.

4.1 Lens Motors(Focus/Zoom) Control Path

Lens Motors(Focus/Zoom) Control Path



4.2 Relationship of Motor Type, PI status, Moving direction and Steps for Zoom and Focus



Relationship of Motor Type, PI status, Moving direction and Steps for Zoom and Focus

Focus Motor Type=0x01 pMC->MotorTypeSet(0x01)

Zoom Motor Type=0x02 pMC->MotorTypeSet(0x02)

Move Motor pMC->MoveMotor()

Infinite Focus

Macro Focus

Zoom Out

Zoom In

pMC->DirectionSet(0x02)
Direction=0x02
Direction to (-)

pMC->DirectionSet(0x01)
Direction=0x01
Direction to (+)

	n	...	3	2	1	1	2	3	....	n	

pMC->MovingStepsSet(n) Steps=n

Steps=n pMC->MovingStepsSet(n)

PI=0 PI=pMC->PIStatus()

PI=1 PI=pMC->PIStatus()

(-)

(+)

CComPtr<EasyLensMotor::IMotorControl> pMC;
pMC.CoCreateInstance(CLSID_EasyLensMotor);

5 IMotorControl interface

IMotorControl provides a basic access with UVC extension unit.

Example: Create EasyLensMotor instance and query a IMotorControl interface

```
CComPtr<EasyLensMotor::IMotorControl> pMC;
hr=pMC.CoCreateInstance(CLSID_EasyLensMotor);
        if(pMC==NULL)
        {
                CString str(_T("\nPlease run RegSvr32 EasyLensMotorDRU.dll first."));
                _tprintf(str);
                ::MessageBox(0,str,_T("EasyLensMotor Error!"),0);
                return;
        }
```

5.1 Assign Device

5.1.1 bool ConnectToDevice(LPCTSTR DeviceName);

Description: connect to device via device name

Example:

```
LPCTSTR DeviceName=_T("USB Video Device");
bool ret=pMC->ConnectToDevice(DeviceName);
if(ret==FALSE) break;
```

5.1 PI Status

5.1.1 int PIStatus(bool bSetPreviousPItoCurrent=false)

Description: get PI status, 0:(-) side, 1:(+) side

Example: Get current PI status, the return value from IsPIStatusChanged() is refer current PI and previous PI status.

```
bool bSetPreviousPItoCurrent=false;

int PI=PIStatus(bSetPreviousPItoCurrent);

if(IsPIStatusChanged()) {/* the PI status changed*/ }
```

Example: Get current PI status and the return value from IsPIStatusChanged() is always false.

```
bool bSetPreviousPItoCurrent=true;

int PI=PIStatus(bSetPreviousPItoCurrent);

if(IsPIStatusChanged()) {/* the PI status changed*/} else {/*PI status not changed*/}
```

5.1.2 bool IsPIStatusChanged(void)

Description: check if the PI status changed, before call this, call PIStatus() first
Return true: when [Current PI status]!= [previous PI status]
Return false: when [Current PI status]== [previous PI status]

5.2 Motor type

5.2.1 bool MotorTypeSet(BYTE val)

Description: assign motor via BYTE value
0x01: focus
0x02: zoom
Return:
true: successful
false: error

Example: assign EasyLensMotor to control the focus motor

```
bool bret=MotorTypeSet(0x01);
```

5.2.2 int MotorType(void)

Description: get motor type value

5.3 Motor Moving Control

5.3.1 Int MoveMotor(void)

Description: call this method to move motor without waiting.

MoveMotor() is to trigger the moving then return immediately, if the moving for long steps, you should set some delay time after call this function.

Example: trigger to move focus to Macro Focus (+) direction with 1000 steps , wait 3 seconds after trigger the moving.

```
MotorTypeSet(0x01);// focus  motor
pMC->MovingStepsSet(1000); // set steps
pMC->DirectionSet(0x01);  // assign direction to (+)
pMC->MoveMotor(); // start move motor

Sleep(3000);
```

5.3.2 int MovingSteps(void)

Description: get single step value
Example:

5.3.3 void MovingStepsSet(int val)

Description: set single step value
Example:

5.3.4 int Direction(void)

Description: get motor moving direction
Return value =0x01, move to (+) direction, for zoom motor means “Zoom In”, for focus motor means “Macro Focus”.
Return value =0x02, move to (-) direction, for zoom motor means “Zoom Out”, for focus motor means “Infinite Focus”.

Example:

5.3.5 void DirectionSet(int dirVal)

Description: assign motor moving direction
dirVal:
0x01, move to (+) direction, for zoom motor means “Zoom In”, for focus motor means “Macro Focus”.
0x02, move to (-) direction, for zoom motor means “Zoom Out”, for focus motor means “Infinite Focus”.

Example: move focus to Macro Focus (+) direction with 10 steps

```
MotorTypeSet(0x01);// focus  motor
pMC->MovingStepsSet(10); // set steps
pMC->DirectionSet(0x01);  // assign direction to (+)
pMC->MoveMotor(); // start move motor
```

6 IMotorControlAdvanced

IMotorControlAdvanced provides some helper functions to reduce application effort.

Example: Create EasyLensMotor instance and query a IMotorControlAdvanced interface

```
CComPtr<EasyLensMotor::IMotorControl> pMC;
CComPtr<EasyLensMotor::IMotorControlAdvanced> pMCA;
hr=pMC.CoCreateInstance(CLSID_EasyLensMotor);
        if(pMC==NULL)
        {
                CString str(_T("\nPlease run RegSvr32 EasyLensMotorDRU.dll first."));
                _tprintf(str);
                ::MessageBox(0,str,_T("EasyLensMotor Error!"),0);
                return;
        }
    hr=pMC->QueryInterface(&pMCA);
```

6.1 bool MotorTypeSetByName(LPCTSTR ZoomOrFocus)

DESCRIPTION: assign motor type via name, it could be "focus" or "zoom"
Example:

```
LPCTSTR sMotorType=_T("Focus");

pMCA->MotorTypeSetByName(sMotorType);
```

6.2 void MotorTypeGetString(LPTSTR ZoomOrFocus)

DESCRIPTION: Helper Functions: get motor type string name
Example:

```
TCHAR ch[255];
RtlZeroMemory(ch,255);
pMCA->MotorTypeGetString (ch);
```

6.3 void MoveMotorGetStatusString(LPTSTR val)

DESCRIPTION: Helper Functions to get motor moving status string
Example:

```
TCHAR ch[255];
RtlZeroMemory(ch,255);
pMCA-> MoveMotorGetStatusString (ch);
```

6.4 void DirectionGetString(LPTSTR val)

DESCRIPTION: Helper Functions: get motor direction readable name.

Example:

```
TCHAR ch[255];
RtlZeroMemory(ch,255);
pMCA->DirectionGetString (ch);
```

6.5 int GetPIStatus(bool bSetPreviousPItoCurrent, int SampleTimes, DWORD PISamplingDelay, double &ratio)

DESCRIPTION: Get PI status with Sampling Times

bSetPreviousPItoCurrent: the same as PIStatus() in “**IMotorControl interface**” chapter, please refer that section
SampleTimes: assign the times to get PI inside **GetPIStatus()**.
PISamplingDelay: assign the delay time(mSecond) for each get PI inside **GetPIStatus().**
ratio: the output ratio for the return value.

Refer below example,
If return value is 0, and ratio=100.0, it means we have 10 times PI=0, => 100% to trust the PI=0
If return value is 0, and ratio=60.0, it means we have 6 times PI=0, 4 times PI=1, => 60% to trust the PI is 0
If return value is 1, and ratio=60.0, it means we have 6 times PI=1, 4 times PI=0, => 60% to trust the PI is 1
If return value is 1, and ratio=90.0, it means we have 9 times PI=1, 1 times PI=0, => 90% to trust the PI is 1

return value:
0:means PI is locate in (-) region
1: means PI is locate in (+) region

Example:

```
bool bSetPreviousPItoCurrent=false;
int SampleTimes=10; //(times)
DWORD PISamplingDelay=50;// mSec
Double ratio=0.0;
Int PI=GetPIStatus(bSetPreviousPItoCurrent, SampleTimes, PISamplingDelay, ratio);
```