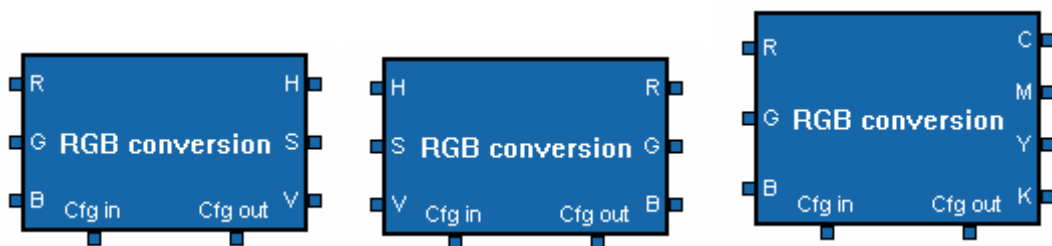


RGB conversion

Obsah

1. Popis.....	1
2. Závislosti	1
3. Implementačné informácie	1
4. Piny	2
5. Konfigurácia	2
5.1 Konfiguračná štruktúra.....	2
5.2 Konfiguračný dialóg.....	4
6. Definícia triedy	4
7. Príklad použitia	6

1. Popis



Trieda *RGB conversion* slúži na konverziu rôznych farebných modelov z alebo do farebného modelu RGB. Trieda podporuje tieto farebné modely: RGB, HSV (HSB), YUV (PAL), YIQ, YCbCr, YCC (Kodak), CMY, CMYK, HLS, CIE XYZ, CIE Lab, HSI, CIE Luv, YUV (SECAM). Vstupom sú tri šedo tónové roviny R, G a B v rovnakom rozlíšení a výstupom sú tri alebo štyri šedo tónové roviny predstavujúce výstupný farebný model (alebo naopak, podľa zvoleného smeru konverzie). Vstupné roviny sú automaticky naškálované do potrebného rozsahu.

2. Závislosti

Trieda *RGB conversion* používa tieto externé definíčné súbory a knižnice:

Headers: *filtergraph.h*, *datatypes.h*

Libs: *filtergraph.lib*

3. Implementačné informácie

Informácie o triede:

Názov: *RGB conversion*

Verzia: 1.0

Magic: 80

Informácie o definíciách:

Header: *rgbconv.h*

Konfiguračný header: *rgbconvcfg.h*

Lib: *rgbconv.lib*

Informácie o knižnici obsahujúcej triedu:

Názov knižnice: *RGB conversion Library*

Verzia knižnice: *1.0*

Dll súbor knižnice: *rgbconv.dll*

4. Piny

Piny sú popísané spôsobom: „*[index pinu na filtri] názov pinu: popis pinu*“.

[0] Cfg in: Konfiguračný vstup. Vstupom je serializovaná konfigurácia.

[1] Cfg out: Konfiguračný výstup. Výstupom je serializovaná konfigurácia.

[2] R: Červený vstup / výstup. Vstupom alebo výstupom je šedo tónová rovina v štruktúre *TPlane*.

[3] G: Zelený vstup / výstup. Vstupom alebo výstupom je šedo tónová rovina v štruktúre *TPlane*.

[4] B: Modrý vstup / výstup. Vstupom alebo výstupom je šedo tónová rovina v štruktúre *TPlane*.

[5] In/Out1: Vstup / výstup. Vstupom alebo výstupom je šedo tónová rovina v štruktúre *TPlane*, podľa zvoleného modelu.

[6] In/Out2: Vstup / výstup. Vstupom alebo výstupom je šedo tónová rovina v štruktúre *TPlane*, podľa zvoleného modelu.

[7] In/Out3: Vstup / výstup. Vstupom alebo výstupom je šedo tónová rovina v štruktúre *TPlane*, podľa zvoleného modelu.

[8] In/Out4: Vstup / výstup. Vstupom alebo výstupom je šedo tónová rovina v štruktúre *TPlane*, podľa zvoleného modelu.

Piny menia svoj typ (vstupný / výstupný) v závislosti od smeru prevodu. Ak prevádzame z modelu RGB, tak piny RGB sú vstupné a piny In/Out sú výstupné. Ak prevádzame do modelu RGB, tak je to opačne. Názvy pinov In/Out sa menia v závislosti od zvoleného farebného modelu. Pin [8] nie je vždy prítomný, je dostupný len v prípade, že zvolený farebný model má štyri zložky (napr. model CMYK). Pri zmene konfigurácie sú piny vytvárané vždy nanovo na daných indexoch.

5. Konfigurácia

5.1 Konfiguračná štruktúra

Konfiguračná štruktúra je definovaná nasledovne:

```
typedef struct {  
    int Conversion;    // konverzia, určuje smer prevodu a farebný model  
} TRGBConversionConfig;
```

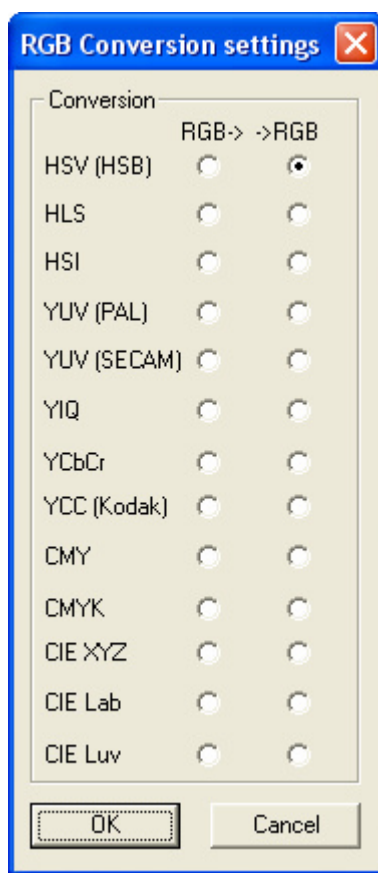
```

// konverzie
CONV_RGBTOHSV      0    // RGB -> HSV
CONV_HSVTORGB      1    // HSV -> RGB
CONV_RGBTOYUV      2    // RGB -> YUV (PAL)
CONV_YUVTORGB      3    // YUV (PAL) -> RGB
CONV_RGBTOYIQ      4    // RGB -> YIQ
CONV_YIQTORGB      5    // YIQ -> RGB
CONV_RGBTOYCBCR     6    // RGB -> YCbCr
CONV_YCBCRTORGB     7    // YCbCr -> RGB
CONV_RGBTOYCC       8    // RGB -> YCC (Kodak)
CONV_YCCTORGB       9    // YCC (Kodak) -> RGB
CONV_RGBTOCMY      10   // RGB -> CMY
CONV_CMYTORGB      11   // CMY -> RGB
CONV_RGBTOCMYK     12   // RGB -> CMYK
CONV_CMYKTORGB     13   // CMYK -> RGB
CONV_RGBTOHLS      14   // RGB -> HLS
CONV_HLSTORGB      15   // HLS -> RGB
CONV_RGBTOXYZ      16   // RGB -> CIE XYZ
CONV_XYZTORGB      17   // CIE XYZ -> RGB
CONV_RGBTOLAB      18   // RGB -> CIE Lab
CONV_LABTORGB      19   // CIE Lab -> RGB
CONV_RGBTOHSI      20   // RGB -> HSI
CONV_HSITORGB      21   // HSI -> RGB
CONV_RGBTOLUV      22   // RGB -> CIE Luv
CONV_LUVTORGB      23   // CIE Luv -> RGB
CONV_RGBTOYUV2     24   // RGB -> YUV (SECAM)
CONV_YUV2TORGB     25   // YUV(SECAM) -> RGB

```

Parameter *Conversion* určuje smer aj farebný model prevodu.

5.2 Konfiguračný dialóg



Konfiguračný dialóg poskytuje voľby popísané pri konfiguračnej štruktúre. V bloku *Conversion* môžeme nastaviť farebný model a smer prevodu, kde stĺpec *RGB->* znamená z modelu *RGB* a stĺpec *->RGB* znamená do modelu *RGB*.

6. Definícia triedy

Trieda *RGB conversion* je definovaná nasledovne:

```
class TRGBConversionFilter : public TFilter {
private:
    HINSTANCE hInstance;           // identifikácia inštancie knižnice

    int Conversion;                // typ konverzie

    int Width;                     // šírka vstupného obrazu
    int Height;                    // výška vstupného obrazu

    TPlane** InPlanes;             // vstupné roviny
    TPlane* OutPlanes;             // výstupné roviny

    TPin* CfgInPin;                // vstupný konfiguračný pin
    TPin* CfgOutPin;               // výstupný konfiguračný pin
    TPin** InPins;                 // vstupné piny
```

```

int cInPins;                // počet vstupných pinov
TPin** OutPins;            // výstupné piny
int cOutPins;              // počet výstupných pinov

bool StopFlag;             // indikátor zastavenia

int RGBtoHSV();             // konverzné funkcie...
int HSVtoRGB();
int RGBtoYUV();
int YUVtoRGB();
int RGBtoYIQ();
int YIQtoRGB();
int RGBtoYCbCr();
int YCbCrtoRGB();
int RGBtoYCC();
int YCCtoRGB();
int RGBtoCMY();
int CMYtoRGB();
int RGBtoCMYK();
int CMYKtoRGB();
int RGBtoHLS();
int HLStoRGB();
int RGBtoXYZ();
int XYZtoRGB();
int RGBtoLAB();
int LABtoRGB();
int RGBtoHSI();
int HSItoRGB();
int RGBtoLUV();
int LUVtoRGB();
int RGBtoYUV2();
int YUV2toRGB();

int MatrixConversion(TMatrix3x3 M, TVec3 rmin, TVec3 rmax);    // maticová metóda

int postScaling();        // škálovanie

int setPins();             // vytvorí vstupno výstupné piny
int freePins();            // uvoľní piny
int freeOutputPlanes();    // uvoľní výstupné roviny

int setConfigFromPin(TPin *pin);    // konfigurácia na pine...
int putConfigOnPin(TPin *pin);
int freeConfigOnPin(TPin *pin);
int clearOutputPins();        // vyčistenie výstupov...
int freeOutputData();

public:
    TRGBConversionFilter();    // konštruktor
    TRGBConversionFilter(HINSTANCE hInst);    // konštruktor s parametrami
    ~TRGBConversionFilter();    // deštruktor

```

```

    int setConfigData(TBuffer config,int type);           // predefinované funkcie triedy TFilter...
    TBuffer getConfigData(int type);
    int initialize();
    int run();
    int reset();
    int stop();
    int finalize();
    int showConfigDialog();
};

```

7. Príklad použitia

