Валерій ФРАТАВЧАН

АЛГОРИТМІЧНІ ОСНОВИ КОМП’ЮТЕРНОЇ ГРАФІКИ

Навчальний посібник для закладів вищої освіти

Чернівці - 2023

УДК 004.92(075.8)

Рекомендовано Вченою радою Навчально-наукового інституту фізико-технічних та комп'ютерних наук Чернівецького національного університету імені Юрія Федьковича (протокол №8 від 22 вересня 2022 року).

Рецензенти:

Барасюк Ярослав – доцент кафедри Інформаційних систем і мереж Чернівецького торговельно-економічного інституту Київського національного торговельно-економічного університету, кандидат фізико-математичних наук;

Рощупкін Олексій Юрійович – кандидат технічних наук, IT компанія «Technology at GlobalLogic, A Hitachi Group Company».

Ф 854 *Фратавчан В.Г.* Алгоритмічні основи комп'ютерної графіки / Навчальний посібник для закладів вищої освіти. – ЧНУ, 2023, – 140 с.

Навчальний посібник містить теоретичні відомості з математичних та алгоритмічних основ інженерної комп'ютерної графіки. Розглядаються математичні моделі та базові алгоритми геометричного перетворення координат, основні математичні перетворення в площинних, стереометричних та однорідних координатах. Аналізуються алгоритми основних стереотипних задач комп'ютерної графіки на площині та у просторі: алгоритми триангуляції, побудови лінійних оболонок, контактної графіки, проведення проєктивних ортогональних та перспективних перетворень. Розглядаються математичні моделі та алгоритми фрактальної графіки та апроксимації ліній та поверхонь вищого порядку. Викладення теоретичного матеріалу супроводжується алгоритмічними схемами та тестовими програмними засобами.

Окремим розділом розглядаються засоби та методи мовної (бібліотечної) графічної надбудови OpenGL. Викладені програмні ресурси для проєктування стереометричних сцен, візуалізації, перетворення, подання зображень у натуральному вигляді.

Навчальний посібник призначений для викладання дисциплін комп'ютерної графіки студентам спеціальностей галузі знань 12 Інформаційні технології.

ЗМІСТ

ПЕРЕДМОВА

Вагома частина інформації, з якою стикається користувач комп'ютерних технологій, має графічну форму. На графічний інтерфейс налаштовані сучасні операційні системи, графічними засобами візуалізується робота системних та прикладних програм, графічними ресурсами насичені інформаційні потоки новин, графічні методи та засоби застосовуються в комп'ютерній ігровій індустрії, графіка масштабно використовується в науковій діяльності, інженерній та конструкторській справах тощо. Сам термін «комп'ютерна графіка» сьогодні трактується як вид діяльності, в якому комп'ютерна техніка та програмне забезпечення використовуються як інструментарій для створення та редагування зображень, для діджиталізації візуальної інформації про реальний світ з метою її подальшої обробки та збереження.

З точки зору розробника комп'ютерних інформаційних та програмних засобів, комп'ютерну графіку можна розділити на дві категорії:

- зображення, які отримуються при використанні пристоїв оптичного введення, а також спеціалізованими графічними редакторами та професійними платформами;
- зображення, які формуються програмним способом.

У даному посібнику основна увага приділяється другій катогорії. В основному ця категорія стосується використання графіки у наукових дослідженнях, в інженерній та конструкторській діяльності, в ігровій індустрії, в професійній діяльності дизайнерського спрямування.

У посібнику розглядається теоретичний математичний інструментарій створення зображень та алгоритми для його програмної реалізації. Посібник містить п'ять розділів, в яких розглядаються відповідні класи задач комп'ютерної графіки:

- опис програмних ресурсів для розв'язування задач комп'ютерної графіки;
- розв'язування задач комп'ютерної графіки та програмне формування зображень на площині;
- розв'язування задач комп'ютерної графіки, моделювання об'єктів та програмне формування зображень у просторі;
- реалізація алгоритмів фрактальної графіки та моделювання стереометричних об'єктів складної нелінійної структури;
- використання бібліотечних мовних засобів для моделювання плоских зображень та просторових об'єктів.

Посібник має форму лекційного курсу, але в ньому врахована орієнтація на алгоритмічну та програмну реалізацію. В багатьох темах, особливо тих, що стосуються базового математичного матеріалу геометричних перетворень та математичних моделей, додатково до теоретичних викладок пропонуються демонстраційні програмні засоби мовами C/C++ (середовище Borland C++ Builder 6.0 використовується виключно з міркувань ергономічності процесу створення програм).

Навчальний посібник призначений для викладання дисциплін інженерної комп'ютерної графіки групи спеціальностей галузі знань «12 - інформаційні технології» (особливо для спеціальності «122-комп'ютерні науки»). Зміст посібника досить деталізований в плані теоретичного математичного матеріалу, але його засвоєння буде легшим при наявності відповідних знань таких розділів вищої математики, як лінійна алгебра та аналітична геометрія. Програмна реалізація описаних у посібнику алгоритмів потребує попередніх знань та навичок програмування.

Автор висловлює вдячність своїм колегам-співробітникам та студентам кафедри математичних проблем управління і кібернетики Чернівецького національного університету імені Юрія Федьковича за цікаві ідеї, допоміжні матеріали, поради та оцінку роботи.

Розділ 1: Графічні засоби мов програмування

Тема 1.1. Керування кольором

1. Колірні моделі

Задачі із застосуванням комп'ютерної графіки умовно можна поділити на дві категорії – задачі з програмної побудови зображень та задачі аналізу та перетворення зображень. За якими алгоритмами ці дії будуть виконуватися та які при цьому будуть задіяні ресурси залежить від формату збереження графічного зображення та від колірної моделі.

Одним з найбільш зручних та «природніх» графічних форматів збереження зображень, а заодно і їх обробки є формат BMP (BitMap Picture). У цьому форматі растрове зображення подається як прямокутна дискретна мапа, на якій кожний піксель має власний колір. Така модель зображення є зручною для позиціювання та локалізації точок зображення, що, в свою чергу, дозволяє успішно використовувати математичний інструментарій для формування, перетворення та обробки зображень. Трактування кольору залежить від задіяної колірної моделі. В комп'ютерній графіці, в залежності від призначення кольорів та від фізичних властивостей, в основному використовуються моделі RBG, CMYK та HSB.

А) Адитивна колірна модель RGB

Дана модель побудована з міркувань апаратної реалізації кольорових комп'ютерних моніторів (за аналогом кольорових телевізорів) на електронно-променевій або рідинно-кришталевій основі трьох базових кольорів – червоного (Red), зеленого (Green), синього (Blue).

Колір елемента зображення визначається 4-байтовим форматом (в мовах C/C++ відповідає типу **longint**). Для кожного базового кольору у числі виділений окремий байт з діапазоном значень яскравості 0-255. Слід зауважити, що четвертий байт числа має власний зміст і у деяких графічних засобах використовується для моделювання «прозорості» середовища.

Таким чином, множину відтінків в колірній моделі RGB можна уявити як куб зі стороною 255 (рис. 1.1.1). Теоретично, ця модель забезпечує можливість оперувати 256^3 кольорами та відтінками (у реальності ця кількість дещо менша).

Кожний колір отримується як «перемішування» базових кольорів в деяких пропорціях, а в математичному розумінні задається лінійною формулою:

$$Col = rR + gG + bB, \tag{1.1.1}$$

де r, g, b – коефіцієнти яскравості відповідних базових кольорів.

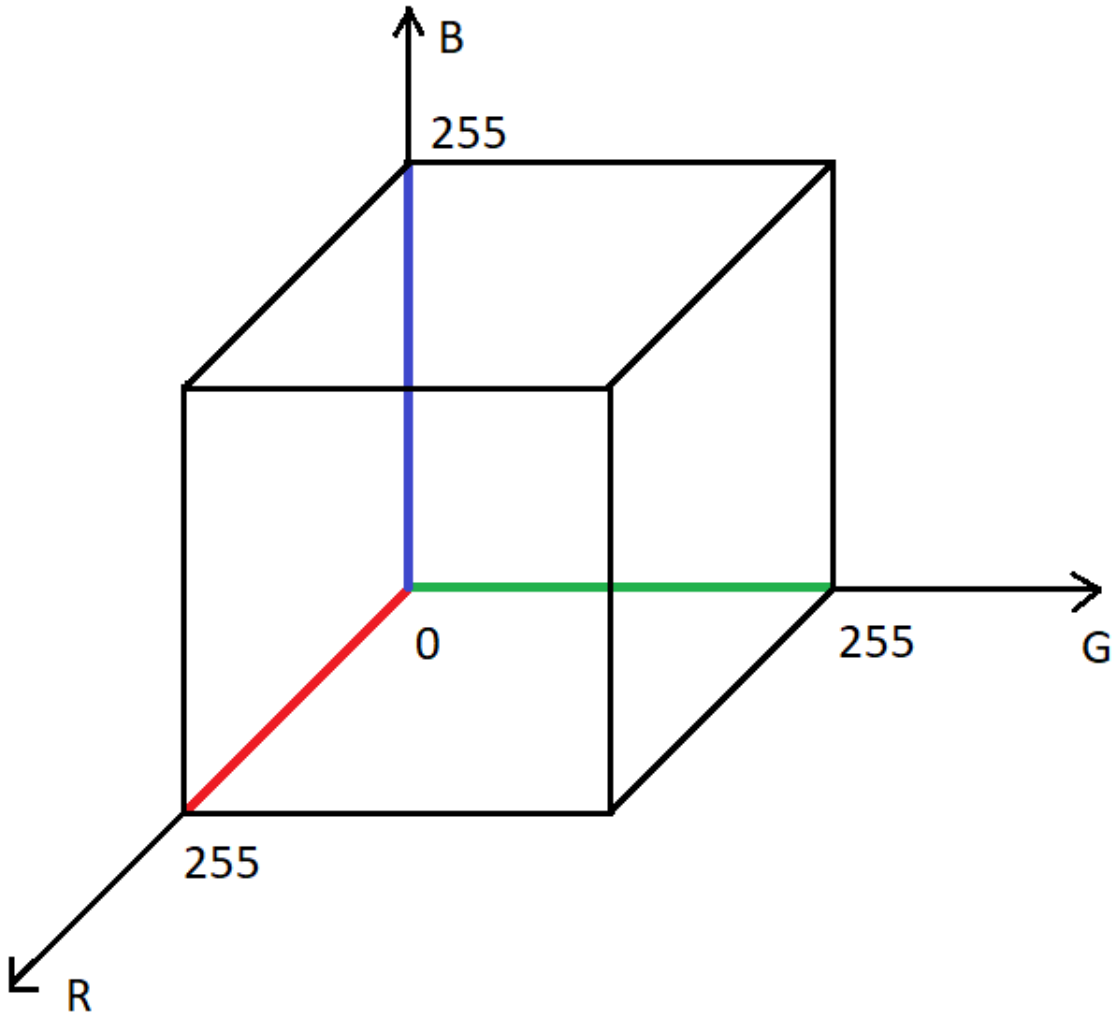


Рис.1.1.1. Адитивна модель кольору RGB.

До цього можна додати, що набір значень базових кольорів (0, 0, 0) відповідає абсолютно чорному кольору (тобто повна відсутність базових кольорів), а набір (255, 255, 255) – абсолютно білому.

Б) Субтрактивна модель кольорів CMY/CMYK

Як відмічалося, модель RGB призначена для екранної комп'ютерної графіки, де сам колір фізіологічно сприймається людиною як безпосередньо

«опромінений». У випадку «поліграфічного», тобто друкованого зображення, колір отримується не внаслідок опромінення, а внаслідок відбиття. Тобто людське око реагує на «залишок» кольору, не поглиненого і відбитого від поверхні білого кольору. А це означає, що базовими кольорами симетричної моделі повинні бути кольори, протилежні до базових кольорів моделі RGB. Такими кольорами є блакитний (Cyan), пурпуровий (Magenta) та жовтий (Yellow). Заголовні букви трьох базових субтрактивних кольорів визначають назву колірної моделі – CMY. «Відбитий» колір можна отримати з «опроміненого» кольору моделі RGB методом віднімання яскравостей базових кольорів з максимальних яскравостей:

$$\begin{pmatrix} C \\ M \\ Y \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} - \begin{pmatrix} R \\ G \\ B \end{pmatrix} \qquad (1.1.2)$$

або, при наших градаціях максимальної яскравості:

$$\begin{pmatrix} C \\ M \\ Y \end{pmatrix} = \begin{pmatrix} 255 \\ 255 \\ 255 \end{pmatrix} - \begin{pmatrix} R \\ G \\ B \end{pmatrix}. \qquad (1.1.3)$$

У моделі CMY є один особливий випадок – модель для друку кольоровим лазерним принтером. При такому друці використовується чотири фарби. До базових кольорів додається чорний колір, якого неможливо «відбити» від білого паперу. Сам друк здійснюється послідовним синхронним чотирикратним друком базових кольорів. При такому підході кожний наступний колір не перемішується, а перефарбовує попередній колір. Тому кожний наступний колір наноситься з непомітним «зсувом» відносно попереднього з коефіцієнтом К. Така модель кольору називається CMYK.

В) Суб’єктивні моделі кольорів HSB та HSV

Адитивна колірна модель RGB чудово підходить для програмного створення зображень але є малоефективними в задачах аналізу та цифрової обробки кольорових зображень, оскільки для двох «сусідніх» відтінків в числовій послідовності чотирибайтових значень кольорів можна отримати істотну різницю візуального сприйняття (залишаємо за Вами міркування про причини цього явища). Тому для аналітичних задач комп’ютерної графіки була запропонована більш «гладка» модель нумерації кольорів та відтінків.

У моделі кольору HSB/V використовуються три «природні» характеристики кольору – тон (Hue), насиченість (Saturation) та яскравість (Brightness/Value). Градації насиченості та яскравості задаються у відносних значеннях 0-1, процентних відношеннях 0-100% або абсолютних значеннях 0-255. Оригінальним для моделі є діапазон тону, який задається у кутових величинах $0\text{-}360^0$. Тобто діапазон видимого колірного спектру розподіляється неперервно по колу у порядку базових кольорів – червоний, оранжевий, жовтий, зелений, блакитний, синій, фіолетовий з поступовим переходом від одного базового кольору до іншого. Такий розподіл кольорів створює візуальний ефект «неперервності» та «гладкості» кольорової гами. Модель легше всього уявити у вигляді циліндра (рис. 1.1.2.):

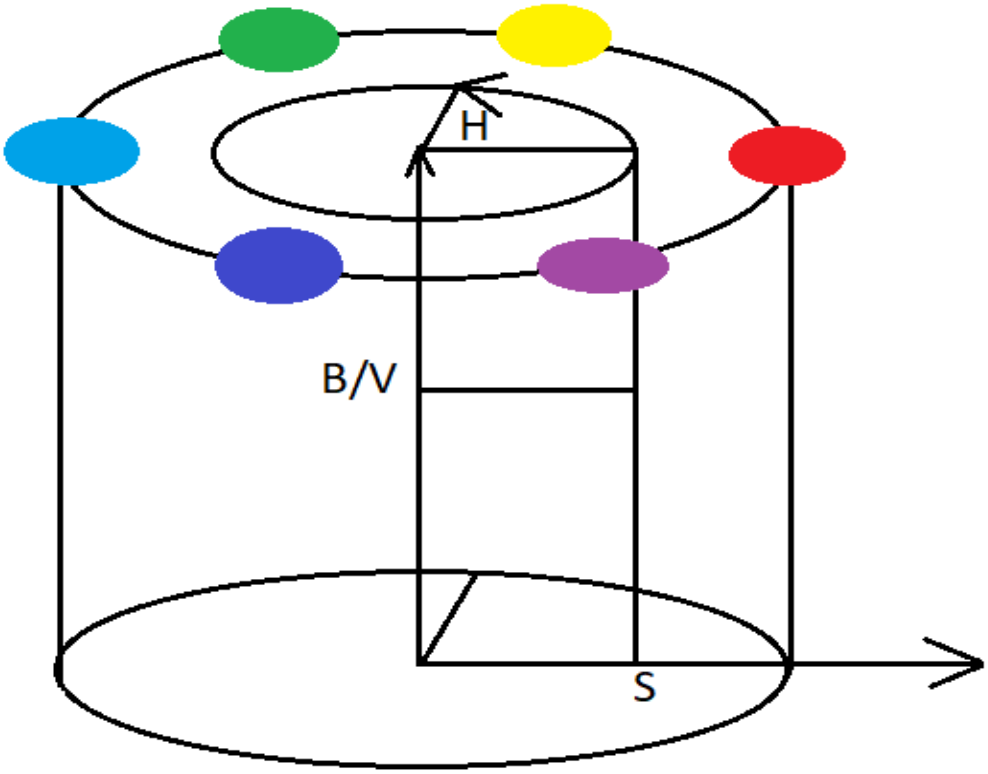


Рис. 1.1.2. Модель кольору HSB/V.

У цьому представлені відмічають високий рівень похибок підчас переходів між HSB/V та RGB моделями при низьких значеннях яскравості. Тому часто застосовують «конічну» форму представлення, в якій горизонтальні перерізи яскравості зверху вниз поступово звужуються до точки в нижній основі циліндра (рис.1.1.3):

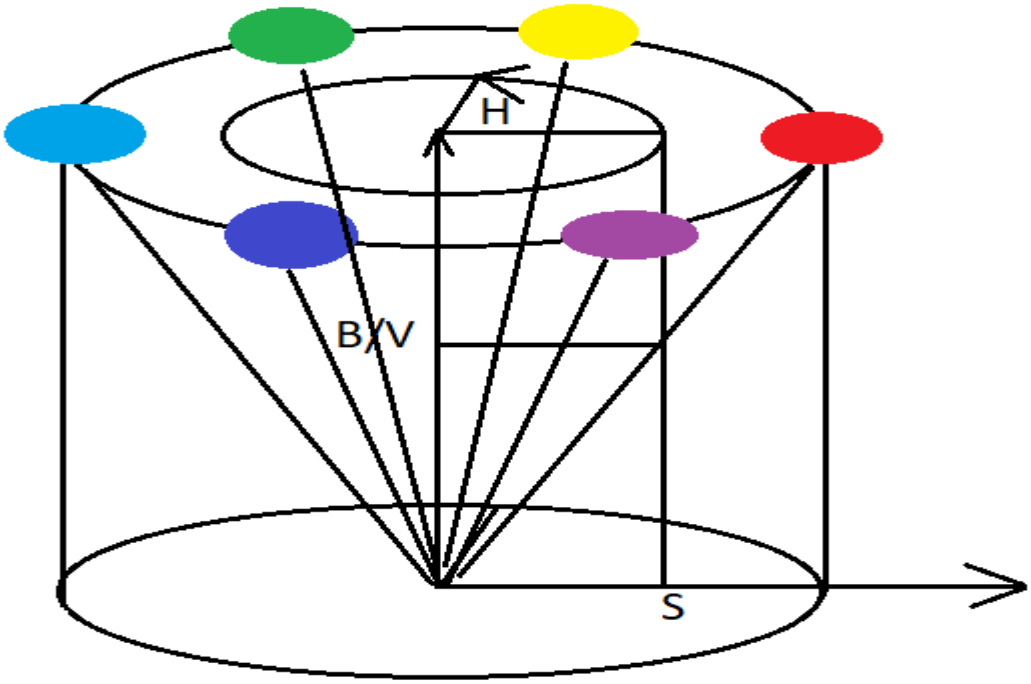


Рис. 1.1.3. Корегована модель кольору HSB/V.

При розгляді колірних моделей потрібно відзначити, що не всі кольори та відтінки, які теоретично передбачені моделлю кольорів, реально доступні в будь-який час. У більшості випадків для графічної платформи вибрана деяка палітра доступних кольорів, яка істотно «бідніша» за максимальні показники моделі. Це пов’язано з технічними характеристиками екранів та моніторів, з крос-платформними питаннями програмних засобів, з протоколами збереження графічної інформації. Тому при роботі з графічними засобами можна бачити обмежені палітри на 16, 256 кольорів тощо.

2. Програмне керування кольором

В задачах на програмну побудову зображень (інженерні креслення, комп’ютерна анімація, комп’ютерна геометрія тощо) керування кольором є необхідним та важливим процесом. В мовах програмування високого рівня

для підтримки комп'ютерної графіки передбачені спеціальні бібліотеки програмних ресурсів.

У інтегрованому середовищі розробки (ICP) Borland C++ Builder (BC++B) для програмування графіки передбачений комплект класів. В класах задані атрибути, інструменти та методи для маніпуляцій графічними об'єктами, для перетворення та формування зображень. Важливим атрибутом цих дій є колір пікселя, лінії, геометричної фігури.

Зауваження. Аналогічний підхід застосовується у більшості сучасних платформ розробки програм із застосуванням комп'ютерної графіки, тому ICP Borland C++ Builder є досить зручним інструментарієм для вивчення, дослідження та демонстрації алгоритмів інженерної комп'ютерної графіки.

Почнемо з того, що ICP Borland C++ Builder зображення можна розмістити безпосередньо на форму (у робочій зоні програмного вікна) або у спеціалізованому компоненті Image (класу TImage). «Інструментами» для програмної побудови зображень служать **точка** (Pixel), **олівець/перо** (Pen), **пензлик** (Brush), які є атрибутами інтегральної властивості «канва» (Canvas) форми та зображення. Тому «керування кольором» при програмуванні зображень можна розуміти як ситуативне призначення кольору для точки, пера, фарби для пензлика за поточними потребами. Для пера та пензлика колір задається властивістю **Color**. А можливості для такого призначення в згаданих засобах програмування досить різноманітні.

А) Використання кольорів з палітри середовища

Як було відмічено у попередньому пункті, окрім «теоретичних» кольорів моделі для графічних платформ може бути визначена і власна палітра кольорів. В ICP BC++B стандартна таблиця має набір кольорів, що задається іменованими константами (рис. 1.1.4).

Constant	Color	Hex value	RGB values		
			Red	Green	Blue
clAqua	Aqua	0x00FFFF00	0	255	255
clBlack	Black	0x00000000	0	0	0
clBlue	Blue	0x00FF0000	0	0	255
clCream	Cream	0x00F0FBFF	255	251	240
clGray	Grey	0x00808080	128	128	128
clFuchsia	Fuchsia	0x00FF00FF	255	0	255
clGreen	Green	0x00008000	0	128	0

clLime	Lime green		0x0000FF00	0	255	0
clMaroon	Maroon		0x00000080	128	0	0
clNavy	Navy		0x00800000	0	0	128
clOlive	Olive green		0x00008080	128	128	0
clPurple	Purple		0x00FF00FF	255	0	255
clRed	Red		0x000000FF	255	0	0
clSilver	Silver		0x00C0C0C0	192	192	192
clTeal	Teal		0x00808000	0	128	128
clWhite	White		0x00FFFFFF	255	255	255

Рис. 1.1.4. Палітра кольорів Borland C++ Builder (частина).

Б) Використання числового значення кольору

Колірна модель RGB теоретично здатна генерувати 256^3 кольорів та відтінків. Для значення кольору використовується формат чотирибайтового беззнакового цілого числа. А це значить, що для точки, олівця, пензлика колір можна призначити аналогічним цілим числом.

Зрозуміло, що значення 3456734 (для прикладу) мало що безпосередньо скаже про колір, але при використанні шістнадцяткових числових констант та при наявності деякого досвіду, заданий колір стає більш прогнозованим. Нагадаємо, що кожна базова складова кольору в послідовності R,G,B займає один байт (при нумерації справа наліво) і задається двома шістнадцятковими цифрами. Числові значення кольорів стандартної палітри BC++B можна побачити на рисунку 1.4 у четвертому стовпчику.

В) **Використання функції генерації кольору *RGB()***

Якщо для поточної роботи кольорів стандартної палітри недостатньо, а шістнадцяткові числові значення створюють складності, для призначення кольорів можна застосувати функцію генерації кольору з трьох її складових:

TColor RGB(int RR, int GG, int BB),

де параметри ***RR, GG, BB*** – значення яскравості червоної, зеленої та синьої базової складової в діапазоні 0-255.

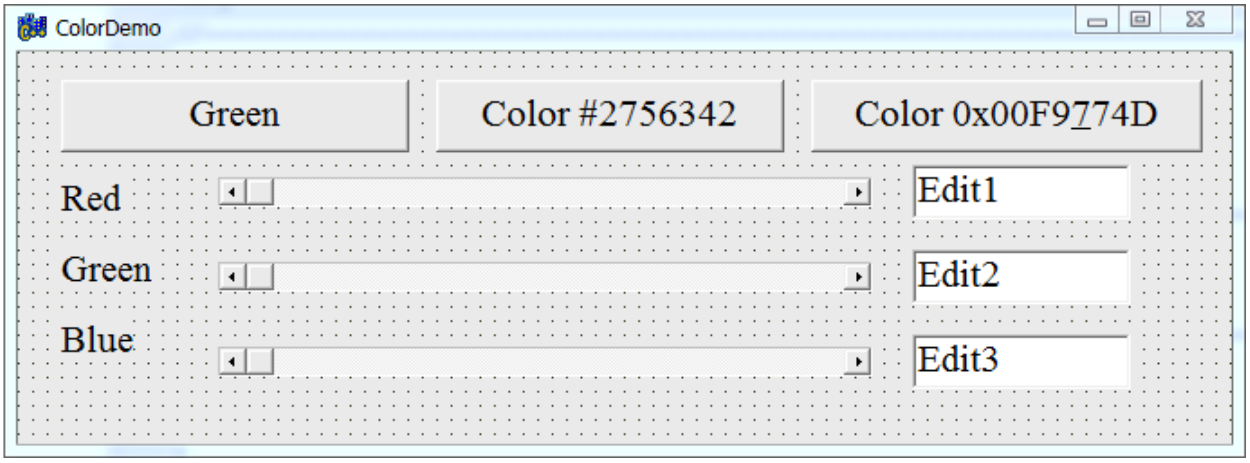
Зауваження. Дана функція є особливо зручною для автоматичної побудови потрібної регулярної систематизованої та обмеженої палітри кольорів.

Як приклад покажемо зразки призначення кольору форми:

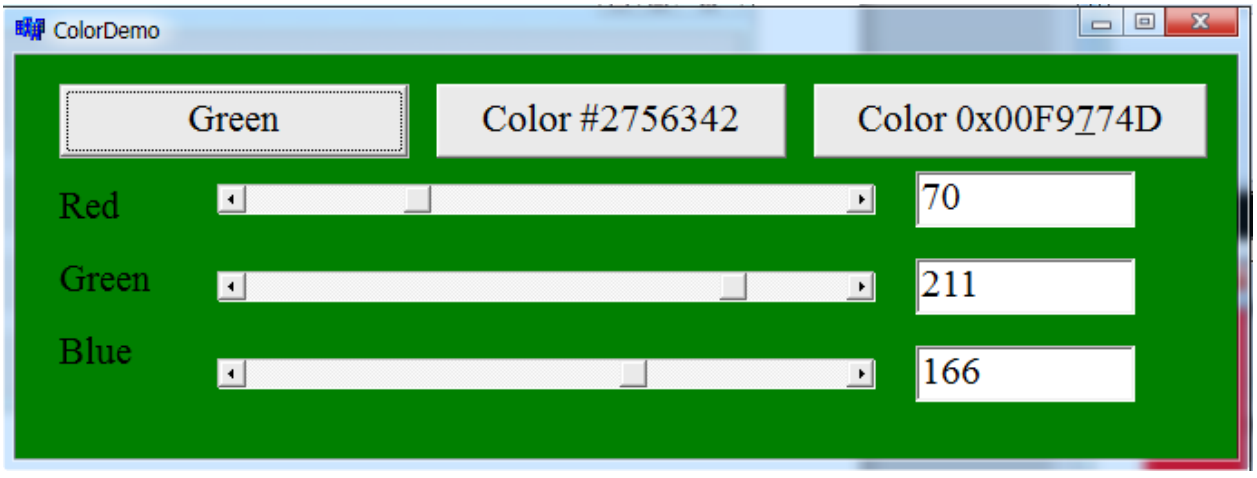
- іменованим значенням:
  Form1->Color = clRed;
- числовим значенням:
  Form1->Color = 56745;
  Form1->Color = 0x5577dd;
- функцією-генератором кольору:
  Form1->Color = RGB(dd,77,55);

3. Демонстраційний приклад керування кольором

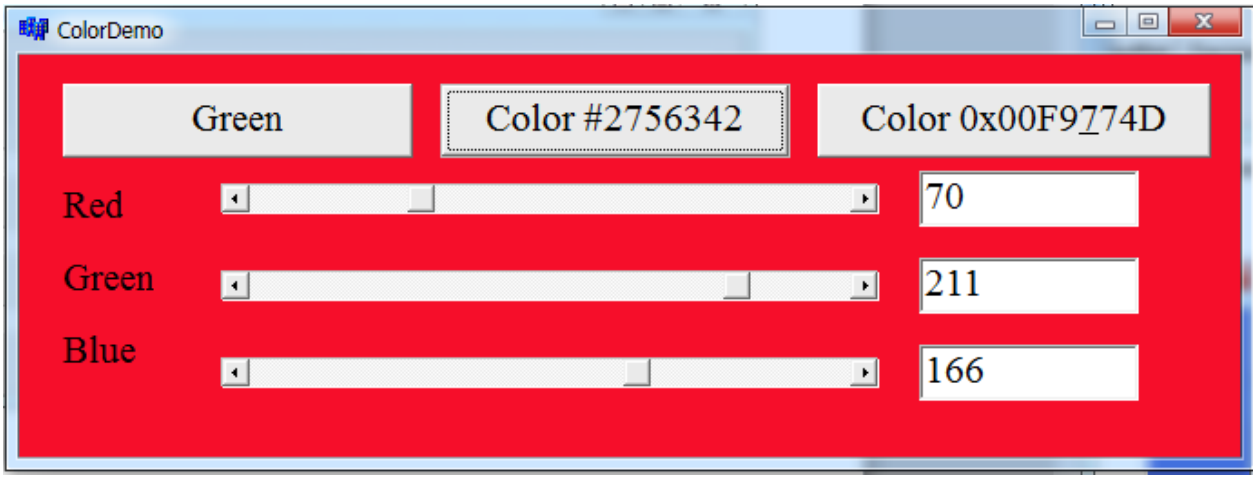
Для візуального огляду описаних вище можливостей керування кольору продемонструємо просту задачу перефарбовування форми. Програмна реалізація не потребує коментарів. Єдине зауваження – компоненти скролінгу генерують числові значення в діапазоні 0-255. У початковому стані форма має сіро-сріблястий колір (рис. 1.5).



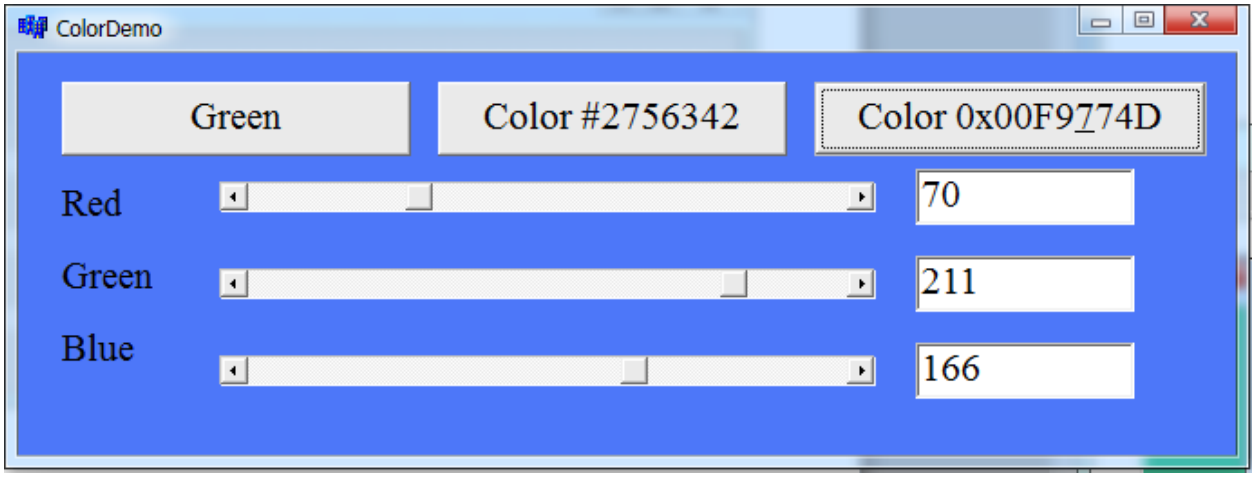
а) Початковий стан форми



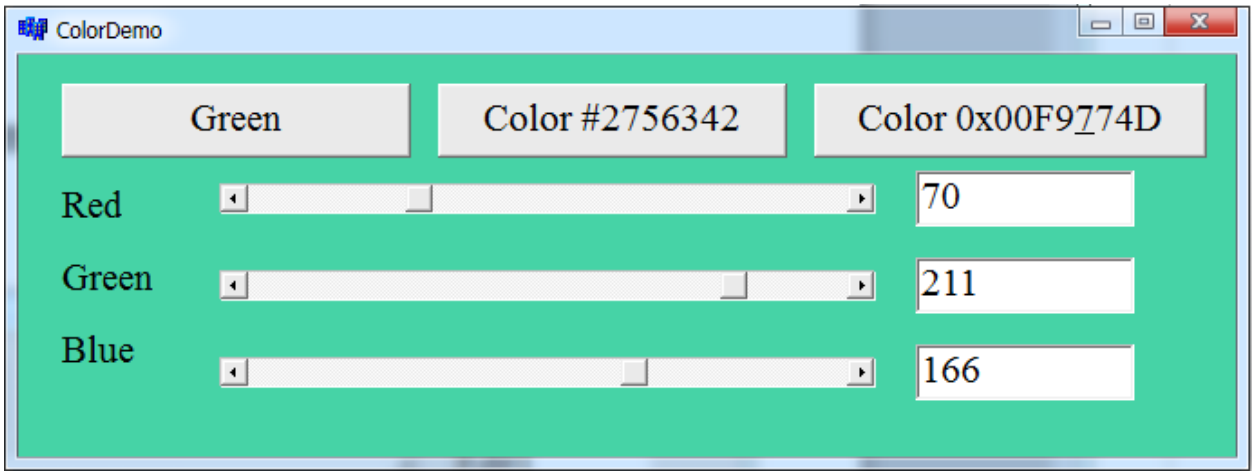
б) Зелений колір форми



в) Форма кольору 2756342



г) Форма кольору 0x00F9774D



д) Колір форми за значеннями лінійок скролінгу

Рис. 1.5. Демонстраційна програма керування кольором

Програмна реалізація проекту:

```
//---------------------------------------------------------------------------
#include <vcl.h>
#pragma hdrstop

#include "Unit1.h"
//---------------------------------------------------------------------------
#pragma package(smart_init)
#pragma resource "*.dfm"
TForm1 *Form1;
//---------------------------------------------------------------------------
__fastcall TForm1::TForm1(TComponent* Owner)
        : TForm(Owner)
{
}
//---------------------------------------------------------------------------
void __fastcall TForm1::Button1Click(TObject *Sender)
{
    Form1->Color = clGreen;
}
//---------------------------------------------------------------------------

void __fastcall TForm1::Button2Click(TObject *Sender)
{
    Form1->Color = 2756342;
}
//---------------------------------------------------------------------------
void __fastcall TForm1::Button3Click(TObject *Sender)
{
   Form1->Color = 0x00F9774D;
}
//---------------------------------------------------------------------------

void __fastcall TForm1::ScrollBar1Change(TObject *Sender)
{   int r,g,b;
   r = ScrollBar1->Position;
   g = ScrollBar2->Position;
   b = ScrollBar3->Position;
```

```
   Form1->Color = RGB(r,g,b);
   Edit1->Text = IntToStr(r);
}
//---------------------------------------------------------------------------
void __fastcall TForm1::ScrollBar2Change(TObject *Sender)
{   int r,g,b;
   r = ScrollBar1->Position;
   g = ScrollBar2->Position;
   b = ScrollBar3->Position;
   Form1->Color = RGB(r,g,b);
   Edit2->Text = IntToStr(g);
}
//---------------------------------------------------------------------------
void __fastcall TForm1::ScrollBar3Change(TObject *Sender)
{    int r,g,b;
   r = ScrollBar1->Position;
   g = ScrollBar2->Position;
   b = ScrollBar3->Position;
   Form1->Color = RGB(r,g,b);
   Edit3->Text = IntToStr(b);
}
//---------------------------------------------------------------------------
```