Санкт-Петербургский национальный исследовательский университет

информационных технологий, механики и оптики

Кафедра информатики и прикладной математики

Проектирование человеко-машинных интерфейсов

Лабораторная работа №1

Выполнил Бакшенов В.О.

Группа P3318

Преподаватель : Зинчик А.А.

2017 г.

Текст Задания

1. Написать программу, выполняющую следующее задание. Для произвольного графического файла формата BMP размером не менее чем 2000x2000 выполнить следующие действия:

1)Загрузить и отобразить файл.

2) Применить действие, согласно варианту задания. Пользоваться функциями GetPixel и SetPixel запрещено.+`

3)Отобразить результат и сохранить новый файл.

2. Оценить отношение скорости работы функции BitBlt и пары функций GetPixel и SetPixel при решении вашего варианта задания и построить график зависимости этого отношения от размеров изображения. Размеры изображения от 200x200 до 2000x2000 с шагом увеличения каждого размера на 200 (200x200, 400x400 и т.д.)
Действие - Убрать синий канал изображения

Текст Программы

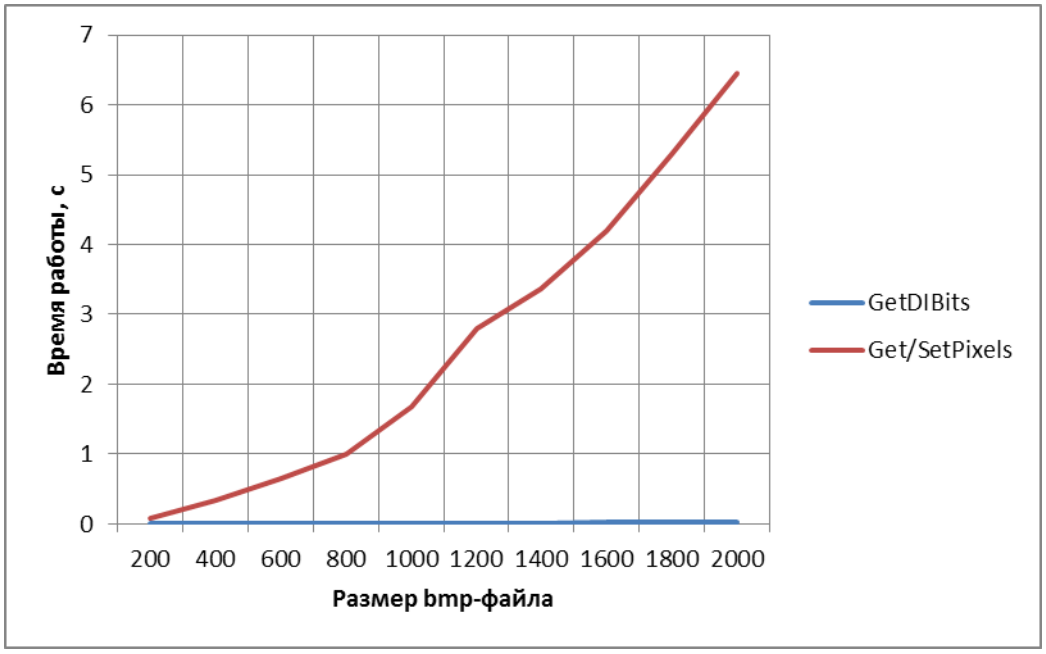
```
int DeleteBlueChannelByBits(HDC hdc, HBITMAP hBitmap, BITMAP bm, double *res, HBITMAP *
hNewBitmap)
{
	DWORD dwBmpSize = ((bm.bmWidth * 32 + 31) / 32) * 4 * bm.bmHeight;

	HANDLE hDIB = GlobalAlloc(GHND, dwBmpSize);
	//lpbitmap - массив байт, куда помещаются цветобайты из изображения
	char *lpbitmap = (char *)GlobalLock(hDIB);
	BITMAPINFO bmi;

	bmi.bmiHeader.biSize = sizeof(BITMAPINFOHEADER);
	bmi.bmiHeader.biHeight = bm.bmHeight;
	bmi.bmiHeader.biWidth = bm.bmWidth;
	bmi.bmiHeader.biPlanes = 1;
	bmi.bmiHeader.biBitCount = 24;
	bmi.bmiHeader.biCompression = BI_RGB;

	//======КЛЮЧЕВАЯ ЧАСТЬ==================

	double start = clock();
	int a = GetDIBits(hdc, hBitmap, 0, (UINT)bm.bmHeight, lpbitmap, (BITMAPINFO
*)&bmi, DIB_RGB_COLORS);

	int size = 3 * bmi.bmiHeader.biHeight*bmi.bmiHeader.biWidth;
	for (int i = 0; i < size;i += 3)
		lpbitmap[i] = 0;

	double finish = clock();
	*res = (finish - start) / CLOCKS_PER_SEC;
	//======КЛЮЧЕВАЯ ЧАСТЬ==================

	//Создаем битмап, в который запишем новый массив rgb-цветов и новый контекст,
чтобы вывести на экран новое изображение
	 *hNewBitmap = CreateDIBitmap(hdc, &bmi.bmiHeader, CBM_INIT, lpbitmap, &bmi,
DIB_RGB_COLORS);

	BITMAPFILEHEADER	bmfHeader;

	HANDLE hFile = CreateFile(TEXT("Result.BMP"),
		GENERIC_WRITE,
		0,
		NULL,
		CREATE_ALWAYS,
		FILE_ATTRIBUTE_NORMAL, NULL);

	DWORD dwSizeofDIB = dwBmpSize + sizeof(BITMAPFILEHEADER) +
sizeof(BITMAPINFOHEADER);
	//Offset to where the actual bitmap bits start.
```

```
	bmfHeader.bfOffBits = (DWORD)sizeof(BITMAPFILEHEADER) +
(DWORD)sizeof(BITMAPINFOHEADER);
	//Size of the file
	bmfHeader.bfSize = dwSizeofDIB;

	//bfType must always be BM for Bitmaps
	bmfHeader.bfType = 0x4D42; //BM

	DWORD dwBytesWritten = 0;
	WriteFile(hFile, (LPSTR)&bmfHeader, sizeof(BITMAPFILEHEADER), &dwBytesWritten,
NULL);
	WriteFile(hFile, (LPSTR)&bmi, sizeof(BITMAPINFOHEADER), &dwBytesWritten, NULL);
	WriteFile(hFile, (LPSTR)lpbitmap, dwBmpSize, &dwBytesWritten, NULL);
	GlobalUnlock(hDIB);
	GlobalFree(hDIB);
	CloseHandle(hFile);
	return 0;
}
```

Используя функции Get/SetPixels

```
//======КЛЮЧЕВАЯ ЧАСТЬ==================

	double start = clock();
	COLORREF obgr;
	for (int i = 0; i < bm.bmHeight; i++)
	{
		for (int j = 0; j < bm.bmWidth; j++)
		{
			obgr = GetPixel(hdc, j, i);// получаем пиксель в формате 0bgr
			obgr = obgr << 16;          // сдвигаем до состояния gr00
			obgr = obgr >> 16;          // сдвигаем назад и получаем 00gr
			SetPixel(hdc, j, i, obgr);
		}
	}
	double finish = clock();
	*res = (finish - start) / CLOCKS_PER_SEC;
//======КЛЮЧЕВАЯ ЧАСТЬ==================
```

Графики



Вывод - Функции Get/Setpixels обрабатывают изображение на порядок медленнее, чем при непосредственной работе с массивом байт.