调色板的颜色混合

stm32

格里特 Feb 16

我们在 CLUT 模式下的 STM32H7 上使用 LV_COLOR_DEPTH 8 的 LVGL。
由于颜色混合器 (lv_color_mix) 不能与调色板一起使用，因此我们实现了一种效果很好的简单机制：
我们需要显示一个简单的 GUI（没有图像），因为所需的基本颜色数量非常少（我们最多使用 32 种颜色）。因此，我们可以使用调色板的其余颜色（最多 224 种）作为预先计算的渐变颜色。我们使用另一个 LUT（有 1024 个条目，每个条目 1 字节）来查找 CLUT 中颜色渐变的索引。

简而言之。如果有兴趣，我们愿意向 LVGL 项目贡献我们的代码。有没有？我们可以创建一个合并请求。

> ✅已解决由 **kisvegabor** 在 **帖子 #9**
>
> 如果使用了清除所有标志的方法，所有用户都需要设置很多标志。现在在某些情况下只需要更改几个标志。我认为它对用户更友好。 请注意，特定的小部件（例如标签）可以设置/清除其自己的标志： 您还可以创建一个像 my_label_create(...

基斯韦加博尔 Feb 16

你好，

每次调用`lv_color_mix()`寻找最佳匹配是不是很慢?

格里特 Feb 16

不，这就是为什么我们有额外的 MixLUT：我们将要混合的两种颜色（每种颜色都是 32 种，所以每种颜色有 5 位）组合成一个 10 位字。这是 MixLUT 的索引，有 1024 个条目，每个条目 1 字节。在这里，我们将找到 CLUT 中预先计算的渐变颜色的第一个条目的索引。

示例：目前我们用 6 种颜色来表示每个颜色渐变，我们称之为颜色分辨率：所以从颜色 c1 到颜色 c2 的颜色渐变如下：

```
100%c1+0%c2  | 80%c1+20%c2 | 60%c1+40%c2 | 40%c1+60%c2 | 20%c1+80%c2 | 0%c1-
      c1     |      c3     |      c4     |      c5     |      c6     |     c
```

跳至主要内容 已在 CLUT 的前 32 个索引中找到，因为它们是基本颜色。我们将混合颜色 c3 到 c6 存储在索引 32 及以上索引中。请记住，颜色只是 CLUT 中存储的 RGB888 颜色的索引。

混合算法的工作原理如下（0<=alpha<=255）：

```
int lv_color_mix(int c1, int c2, int alpha) {
  if (alpha==255)
    return (c1);
  if (alpha==0)
    return (c2);
  idx=MixCLUT[c1<<5|c2];
  return (idx+((alpha*6)>>8)-1);
}
```

基斯韦加博尔 Feb 16

我明白了，谢谢你的解释。但有一件事还不清楚。如何制作 LUT?

前 32 种颜色是 UI 的颜色。天气晴朗。其余的应该分为 4 种颜色

1. 80%c1+20%c2
2. 60%c1+40%c2
3. 40%c1+60%c2
4. 20%c1+80%c2

因此，`MixCLUT` 可以指向 (256-32) / 4 = 56 个“入口点”（4 种颜色从 4 alpha 值开始）。因此您需要将 32x32 = 1024 种颜色组合映射到 56 个索引。 （我看到有`20%c1 + 80%c2` 和`80%c2 + 20%c1`一样，但还是很远）。

有什么窍门呢? 🙂

格里特 Feb 16

是的，MixLUT 并不存储所有理论上可能的颜色渐变。但至少在我们的例子中，设计也不需要它 - 按钮的圆角，例如始终以相同的颜色绘制，但使用 3 种不同的背景颜色。所以我们需要 3 种颜色渐变，或者 12 种颜色。我们还需要其他颜色渐变，但当然我们需要将其保持在最多 56 个颜色渐变以下。当我们只想管理一半的帧缓冲区大小时，这种限制是我们必须付出的代价……

基斯韦加博尔 Feb 17

我明白了。对我来说，这似乎是一个非常定制的解决方案，我几乎无法想象有一种通用的方式来支持它。我认为最简单且更通用的方法是允许覆盖 `lv_color_mix()` 例如添加 `#if LV_COLOR_MIX_EXTERNAL`左右。

你怎么认为?

格里特 Feb 17

跳至主要内容

是的，带有 CLUT 的颜色渐变可能有点具体。我们已经根据具体的 UI 设计考虑了许多选项……目前看来，限制最多 57 种渐变比限制 CLUT 中最多 256 种颜色更容易处理。
新的微控制器即将问世，它们具有足够的内部存储器，可以以 8bpp 的速度存储完整的 800x480 像素帧缓冲区，例如 STM32H7。这降低了系统成本，并可能增加人们对更“通用”解决方案的兴趣。但就目前而言，我认为您拥有定制混色器的想法是最好的 - 我们将不胜感激。作为建议，我们提交了一份涵盖我们要求的拉取请求。请看一下：

feat(color)：添加了覆盖 lv_color_mix 的选项

lvgl:master←domologic:color_mix

2023 年 2 月 17 日 家庭学 +77 -38

功能或修复的描述

清晰简洁地描述内容 ...

我们还需要覆盖 lv_color_premult 和 lv_color_mix_premult。

格里特 Feb 17

另一个问题：在我们的例子中，在 lv_obj_constructor() 中完成的基础对象的初始配置不太合适，因此我们必须取消屏蔽一些标志，例如 LV_OBJ_FLAG_CLICKABLE 或 LV_OBJ_FLAG_SNAPPABLE，因为不可能拖动标签。

在不设置所有标志的情况下初始化基础对象，并在每个小部件中单独设置它们不是更好吗？这也会更有效一些，因为默认情况下不会设置这些位，并且必须在小部件中重置，正如我们在某些情况下看到的那样。

基斯韦加博尔 Feb 20

如果使用了清除所有标志的方法，所有用户都需要设置很多标志。现在在某些情况下只需要更改几个标志。我认为它对用户更友好。

请注意，特定的小部件（例如标签）可以设置/清除其自己的标志：

lvgl/lvgl/blob/master/src/widgets/label/lv_label.c#L678

```
668     label->hint.y           = 0;
669 #endif
670
671 #if LV_LABEL_TEXT_SELECTION
672     label->sel_start = LV_DRAW_LABEL_NO_TXT_SEL;
673     label->sel_end   = LV_DRAW_LABEL_NO_TXT_SEL;
    #endif
```

跳至主要内容

```
675     label->dot.tmp_ptr   = NULL;
676     label->dot_tmp_alloc = 0;
677 
678     lv_obj_clear_flag(obj, LV_OBJ_FLAG_CLICKABLE);
679     lv_label_set_long_mode(obj, LV_LABEL_LONG_WRAP);
680     lv_label_set_text(obj, LV_LABEL_DEFAULT_TEXT);
681 
682 
683     LV_TRACE_OBJ_CREATE("finished");
684 }
685 
686 static void lv_label_destructor(const lv_obj_class_t * class_p,
687 {
688     LV_UNUSED(class_p);
```

您还可以创建一个类似 `my_label_create(lv_obj_t * parent)` 的函数，它创建一个普通标签并更改一些标志。

格里特 Feb 25

谢谢您，我想我可以遵循您的方法。
也感谢您合并我们的 lv_color_mix 拉取请求 🙂