HW3

`opencv`

---

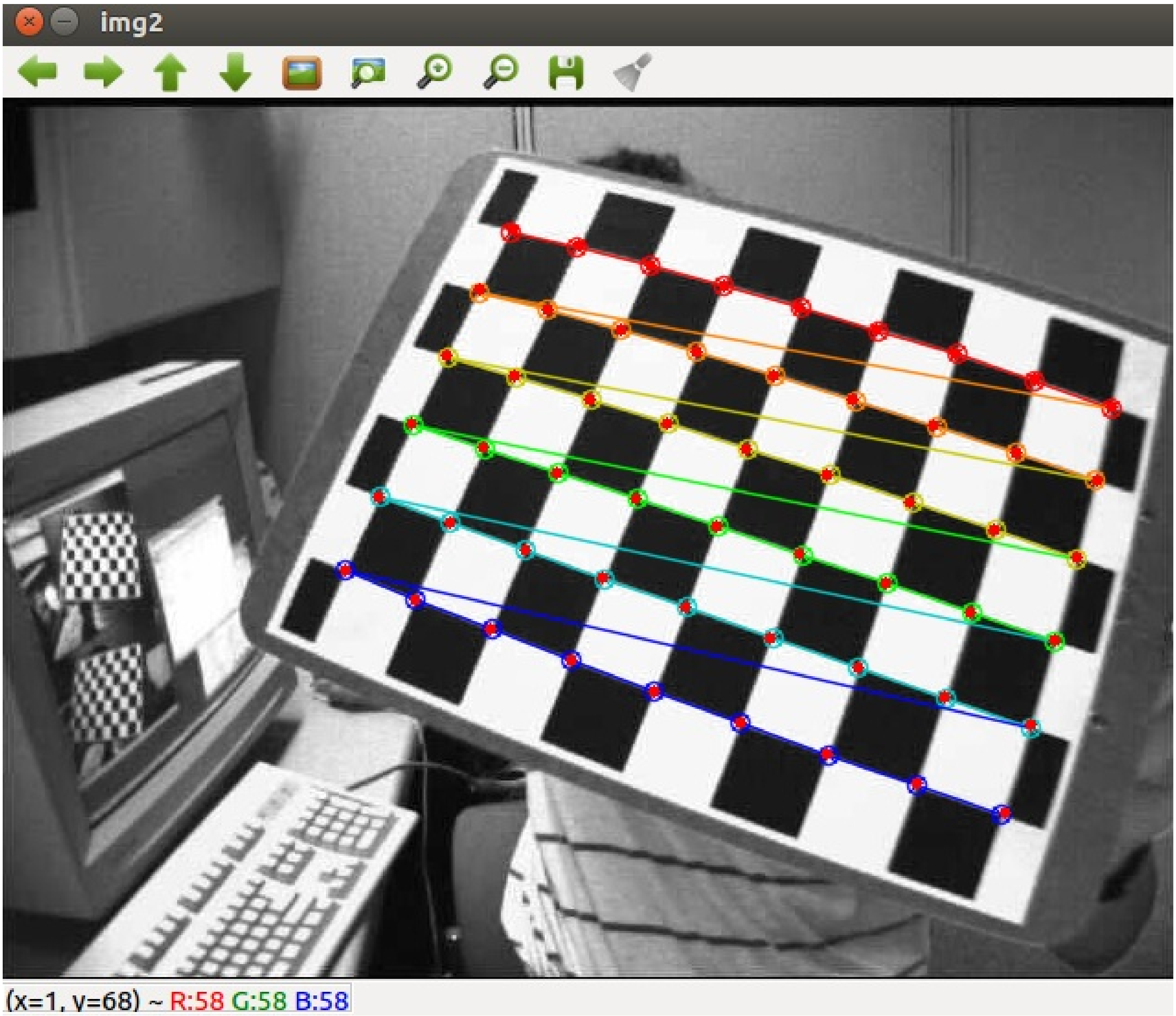
1 代码执行和演示视频

- 作业第一题对应find_homography.py文件
- 作业第二题对应guess_num.py文件

两个作业都有对应的演示视频。

2 计算单应矩阵

下图是通过计算出1图到2图的H后，直接用1图中的角点左乘H，得到2图中的角点做的一个验证。可以看到2图中红色点正好和角点重合。



3 算法题解决思路和问题描述

一个数字为n，在n中任意选一个数字作为 $answer$ ，可以一直猜测下去，直到猜对。如果用$x \in [1, n]$表示该次使用的猜测数字，如果猜错，则猜错的代价$cost(x) = x$。

所以对于 n 中的其中一个 $answer$ ，我们如果要猜对的话，那么可能需要猜测很多次，假设我们采用的某种猜测策略需要猜测m次,每次猜的数字为x_i。那么为了猜对这个 $answer$ ，我们的m次猜测错误代价总和表示为

$$answer : \sum_{i=1}^{m} cost(x_i) = answer : \sum_{i=1}^{m} x_i$$

考虑一般情况，在区间$[j,i], 1 <= j < i <= n$中，如果我们使用类似分治法的思想，在区间中任取k作为猜测值，那么这个区间内的某一个$answer$的代价包括三个部分，k，k的左区间和k的右区间的代价和，表示为

$$answer : cost(k) + max(cost(k_{left}), cost(k_{right}))$$

也就是针对每一次猜测分割用一个**局部最大值(local_max)**加上k表示能够覆盖本次猜测的所有可能消耗的代价。

但是区间$[j,i]$中包括了$i-j$种 k 的可能，所以我们需要遍历这$i-j$种可能，分别求出对应的代价，表示为

$$cost_j, \ldots, cost_i$$

要注意，在区间$[j,i]$上计算代价时有两种可能，第一次取k作为猜测的分割点

- 如果$i-j=1$,那么直接求出区间代价为j,
- 如果$i-j>=2$,那么从$k=j+1$到$k<=i-1$遍历，然后在所有的代价中取一个最小值，表示为$\min \{cost_j, \ldots, cost_i\}$, 也称为**全局最小值(global_min)**, 即为此区间代价。

现在还需要解决最后一个问题，如何保证我们前面的$cost(k_{left}), cost(k_{right})$是知道的？

需要两点保证
(1)总区间$[1,n]$,采用如下的遍历方式，

```
for i in xrange(2,n+1,1):
    for j in xrange(i-1,0,-1):# 先求[j,i]区间内的cost
```

(2) 相差为1的区间代价计算方式
前面提到的，**如果$i-j=1$,那么直接求出区间代价为j。**
如上两点保证后面用到的每一个区间代价，都是经过前面计算的，也即是动态规划思想，利用前面计算的结果来计算当下的问题。