**数据的处理方式** 这种数据获取方式比较明朗看到数据内部结构 【<http://blog.csdn.net/glb562000520/article/details/39989995?utm_source=tuicool&utm_medium=referral>】

1. INuiFrameTexture \*pTexture =pImageFrame->pFrameTexture;
2. NUI\_LOCKED\_RECT LockedRect;
3. pTexture->LockRect(0, &LockedRect,NULL, 0);
5. RGBQUAD q;
7. **if**( LockedRect.Pitch != 0 )
8. {
10. //BYTE \* pBuffer = (BYTE\*)(LockedRect.pBits);
11. //INT size =  LockedRect.size;
12. //memcpy\_s(m\_pDepthBuffer,size, pBuffer, size);
13. //USHORT\* pBufferRun =reinterpret\_cast<USHORT\*>(m\_pDepthBuffer);
14. **for** (**int** i=0; i<image.rows; i++)
15. {
16. //USHORT\* ptr = (USHORT\*)depthIndexImage->height;
17. //USHORT\* pDepthRow =(USHORT\*)(i);
18. //BYTE \* pBuffer = (BYTE\*)(LockedRect.pBits);
19. uchar \*ptr =image.ptr<uchar>(i);  //第i行的指针
20. uchar \* pBuffer =(uchar\*)(LockedRect.pBits)+i\*LockedRect.Pitch;
21. **USHORT**\* pBufferRun =(**USHORT**\*) pBuffer;//注意这里需要转换，因为每个数据是2个字节，存储的同上面的颜色信息不一样，这里是2个字节一个信息，不能再用BYTE，转化为USHORT
23. **for** (**int** j=0; j<image.cols; j++)
24. {
25. //ptr[j] = 255 -(BYTE)(256\*pBufferRun[j]/0x0fff);//直接将数据归一化处理
26. //ptr[j]  = pBufferRun[i \* 640 + j];
27. // ptr[j] = 255 -(uchar)(256 \* pBufferRun[j]/0x0fff);  //直接将数据归一化处理
28. **int** player =pBufferRun[j]&7;
29. **int** data =(pBufferRun[j]&0xfff8) >> 3;
31. uchar imageData = 255-(uchar)(256\*data/0x0fff);
32. q.rgbBlue = q.rgbGreen =q.rgbRed = 0;
34. **switch**(player)
35. {
36. **case** 0:
37. q.rgbRed = imageData /2;
38. q.rgbBlue = imageData / 2;
39. q.rgbGreen = imageData/ 2;
40. **break**;
41. **case** 1:
42. q.rgbRed =imageData;
43. **break**;
44. **case** 2:
45. q.rgbGreen =imageData;
46. **break**;
47. **case** 3:
48. q.rgbRed = imageData /4;
49. q.rgbGreen = q.rgbRed\*4;  //这里利用乘的方法，而不用原来的方法可以避免不整除的情况
50. q.rgbBlue =q.rgbRed\*4;  //可以在后面的getTheContour()中配合使用，避免遗漏一些情况
51. **break**;
52. **case** 4:
53. q.rgbBlue = imageData /4;
54. q.rgbRed = q.rgbBlue\*4;
55. q.rgbGreen =q.rgbBlue\*4;
56. **break**;
57. **case** 5:
58. q.rgbGreen = imageData/ 4;
59. q.rgbRed =q.rgbGreen\*4;
60. q.rgbBlue =q.rgbGreen\*4;
61. **break**;
62. **case** 6:
63. q.rgbRed = imageData /2;
64. q.rgbGreen = imageData/ 2;
65. q.rgbBlue =q.rgbGreen\*2;
66. **break**;
67. **case** 7:
68. q.rgbRed = 255 - (imageData / 2 );
69. q.rgbGreen = 255 - (imageData / 2 );
70. q.rgbBlue = 255 - (imageData / 2 );
71. }
72. ptr[3\*j] = q.rgbBlue;
73. ptr[3\*j+1] = q.rgbGreen;
74. ptr[3\*j+2] = q.rgbRed;
75. }
76. }
78. imshow("depthImage",image); //显示图像
79. 得到的最终形式可以用OpenCV显示。