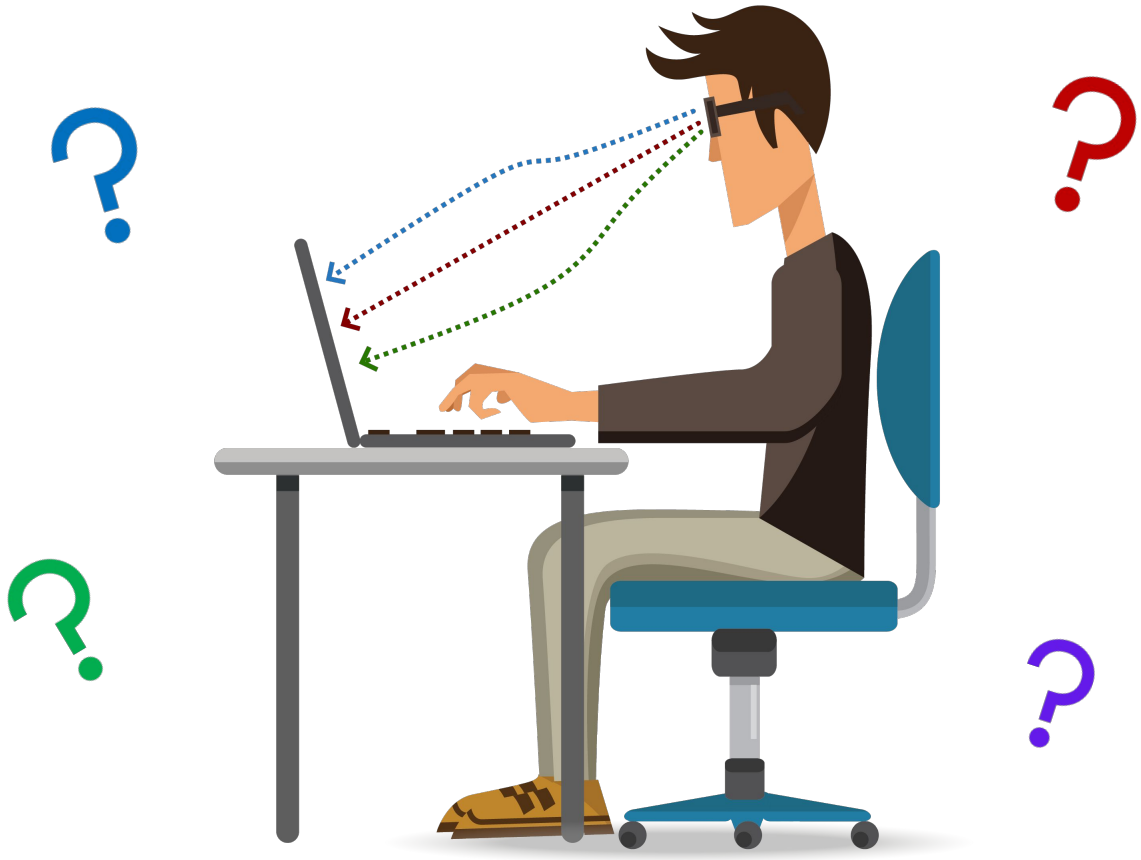
Eye Tracking, разработка алгоритма определения зрачка на растровом изображении

РАБОТУ ВЫПОЛНИЛ: УЧАСТНИК ЦМИТ "ZABLAB",
ГОРДЕЕВ ИВАН ЕВГЕНЬЕВИЧ
НАУЧНЫЙ РУКОВОДИТЕЛЬ: ИНСТРУКТОР РОБОТОТЕХНИКИ ЦМИТ "ZABLAB",
ДАВЫДОВИЧ АЛЕКСАНДР ЕВГЕНЬЕВИЧ

Что такое Eye Tracking?



Где это может применяться?

Маркетинговые исследования

Виртуальная реальность

Где это может применяться?

Коммуникационные системы для полностью парализованных людей

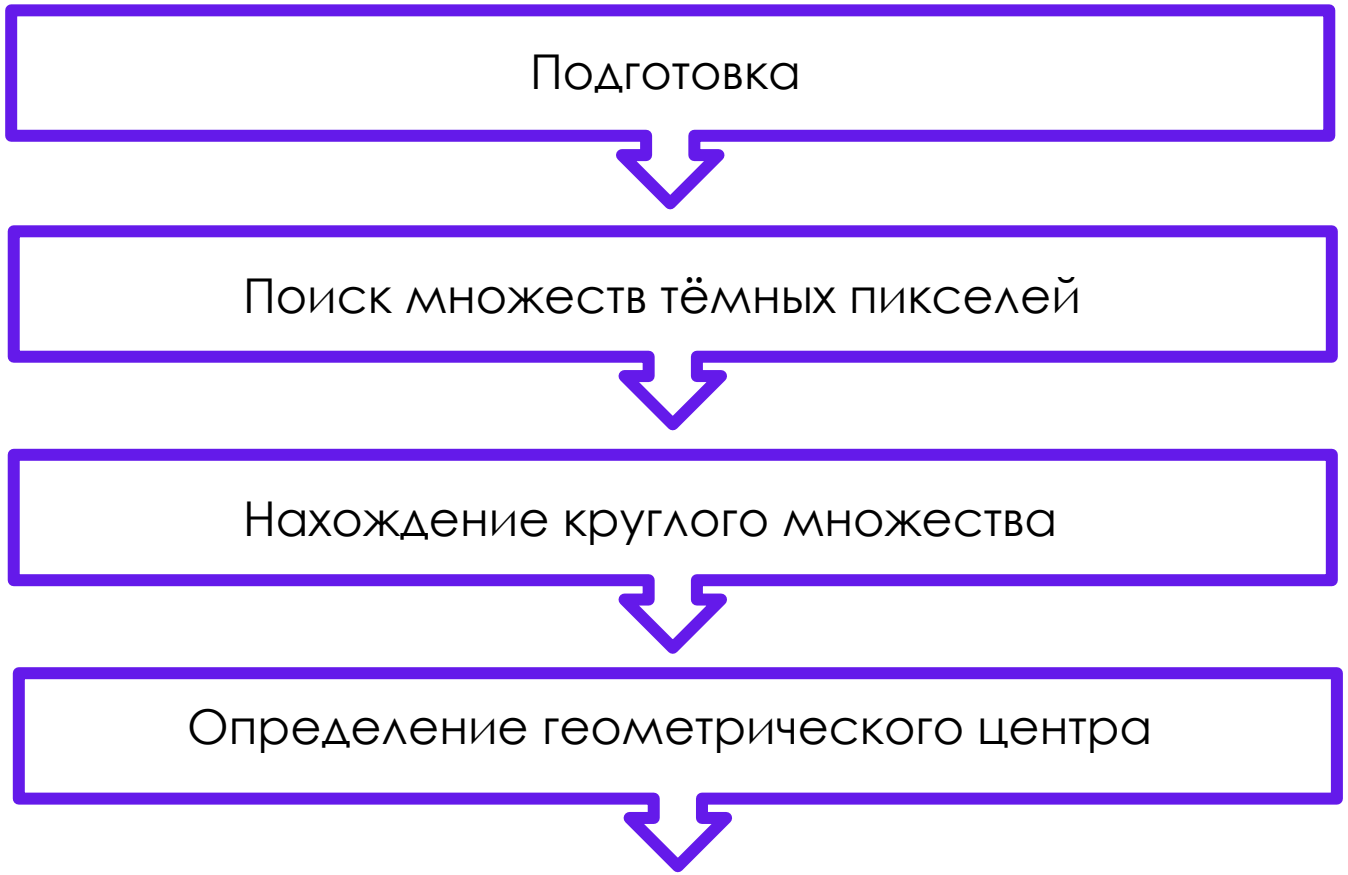
Управление компьютером

Аппаратная часть

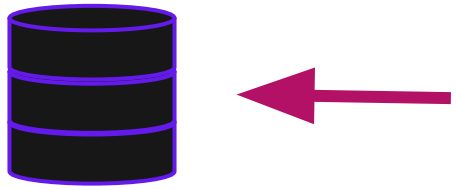
Веб-камера, без ИК-фильтра

Инфракрасная подсветка

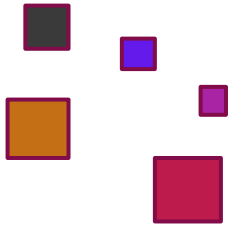
Этапы работы алгоритма



Этап 1: Подготовка



Оперативная память

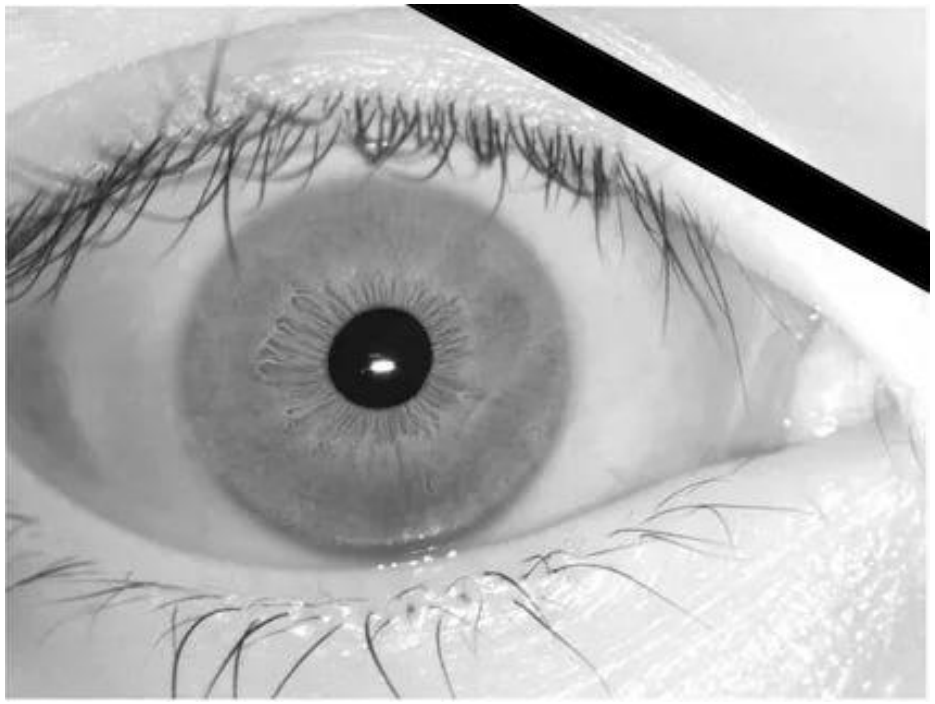


Цвета всех пикселей изображения, полученного с веб-камеры

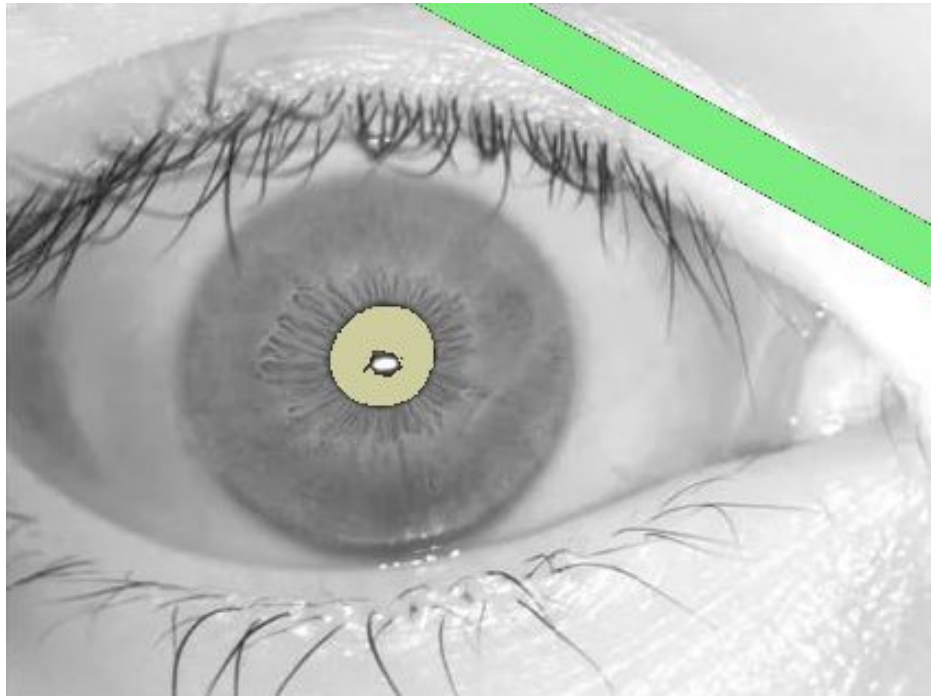
Исходный код (фрагмент):

```
Bitmap bitmap = Camera.Frame;
BitmapData bitmapData = bitmap.LockBits(new Rectangle(0, 0,
bitmap.Width, bitmap.Height),
    ImageLockMode.ReadOnly, bitmap.PixelFormat);
IntPtr ptr = bitmapData.Scan0;
int count = Math.Abs(bitmapData.Stride) * bitmap.Height;
byte[] colors = new byte[count];
bool[,] flags = new bool[bitmap.Height,bitmap.Width];
System.Runtime.InteropServices.Marshal.Copy(ptr, colors, 0,
count);
        bitmap.UnlockBits(bitmapData);
        List<List<Point>> groups = new List<List<Point>>();
        List<List<Point>> glints = new List<List<Point>>();
        List<Point> barycentersOfGlints = new List<Point>();
```

Этап 2: Поиск множеств тёмных пикселей



Входные данные алгоритма



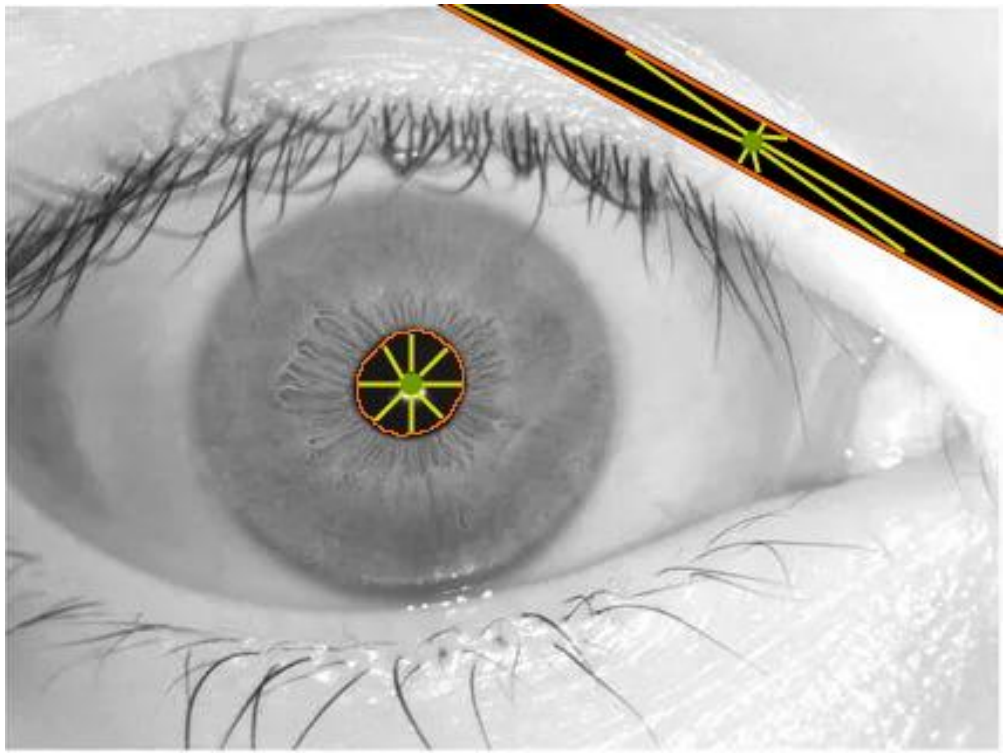
Выходные данные 2ого этапа выполнения алгортима

Этап 2: Поиск множеств тёмных пикселей

Исходный код перебора всех пикселей

```
for (int i = 3; i < bitmap.Height - 3; i += 3)
        {
           for (int j = 9; j < bitmapData.Stride - 9; j += 9)
           {
              int R = colors[i * Math.Abs(bitmapData.Stride) + j];
              int G = colors[i * Math.Abs(bitmapData.Stride) + j + 1];
              int B = colors[i * Math.Abs(bitmapData.Stride) + j + 2];
if (R <= PUPIL_COLOR_BORDER &&
G <= PUPIL_COLOR_BORDER && B <= PUPIL_COLOR_BORDER)
              {
                 Point p = new Point(j / 3, i);
                 if (!flags[p.Y, p.X])
                 {
                    List<Point> points = FindGroup(p);
                    if (points.Count >= 500)
                       groups.Add(points);
                 }
              }
```

Этап 3: Нахождение круглого множества



```
bool IsRound(List<Point> l, out double dif)
 {
            double maxDistance = -1.0;
            double radius = 0;
            Point center = GetCenter(l);
            foreach (Point p in l)
            {
               double currentDistance = GetDistance(p, center);
               if (currentDistance > maxDistance)
                  maxDistance = currentDistance;
               radius += currentDistance;
            }
            radius /= l.Count;
            double[] radiuses = GetRadiuses(l, int leftX, int upY);
            dif = Math.Max(Math.Max(Math.Abs(radiuses[0] -
radiuses[1]), Math.Abs(radiuses[0] - radiuses[2])),
Math.Abs(radiuses[1] - radiuses[2]));
            return true;
}
```

Этап 4: Определение барицентра

Функция вычисления барицентра:

```
Point GetCenter(List<Point> p)
        {
            int avrX = 0, avrY = 0;
            for (int i = 0; i < p.Count; i++)
            {
                avrX += p[i].X;
                avrY += p[i].Y;
            }
            return new Point(avrX / p.Count, avrY / p.Count);
        }
```

Спасибо за внимание!