Разработка системы отслеживания взгляда чел👀века

Французяк Ярослав

веб-разработчик

Как появилась идея

и в чем ее актуальность

Типовые решения

	Внешняя приставка, подключаемая к устройству	**Webgazer**
Достоинства	Готовое аппаратное и программное обеспечение, высокая точность	Мобильное open source решение
Недостатки	Внешняя приставка, подключаемая к устройству	Сложная калибровка, зависимость от положения головы

Зачем создавать свою систему?

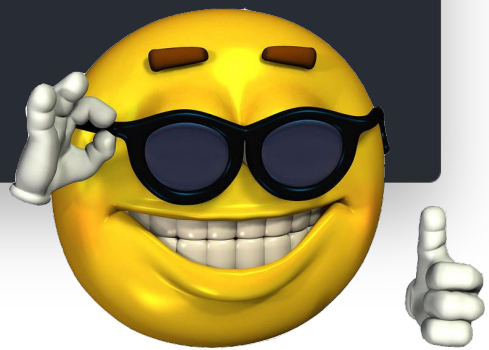
Всё можно сделать лучше, чем делалось до сих пор.

Следует взять что-либо, доказавшее свою пригодность, и устранить в нем все лишнее.

В великой мудрости есть великая горесть, и кто умножит знание — умножит скорбь.

Функциональное декларативное решение

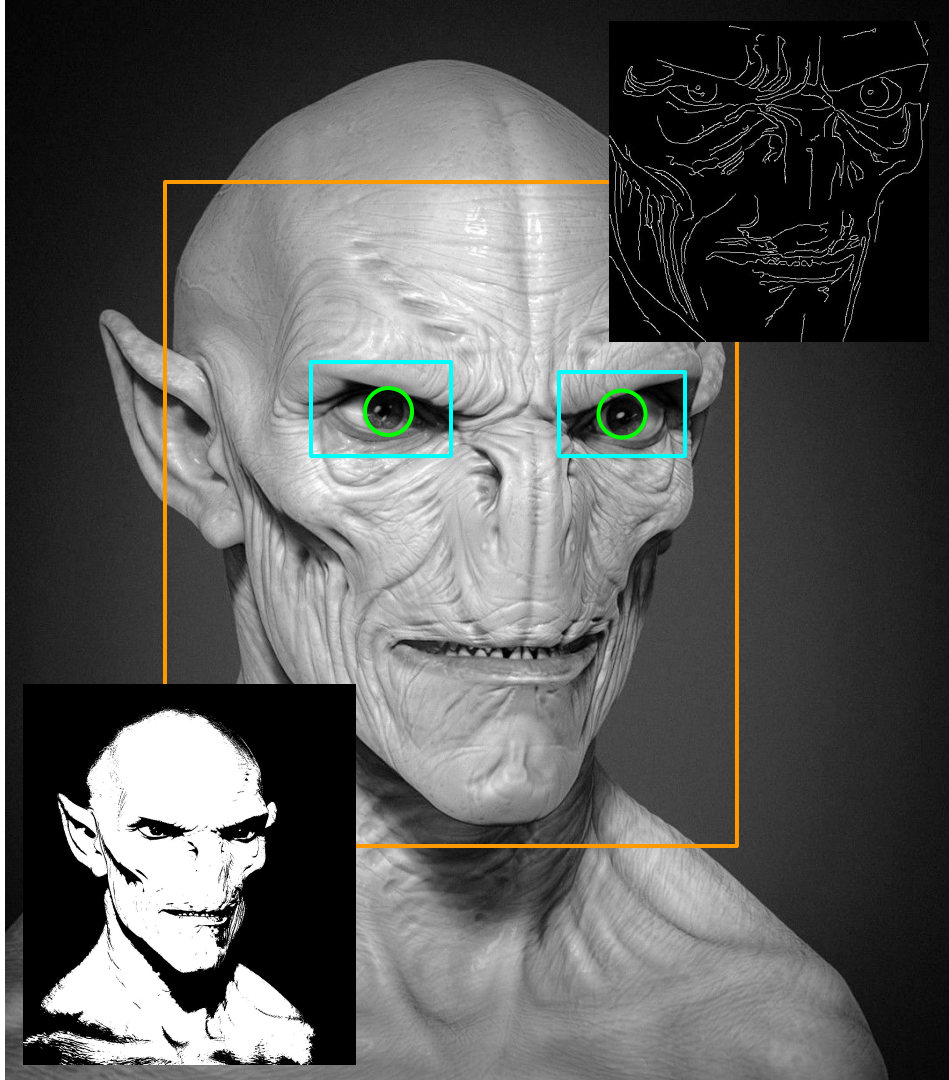
```
получитьВидеопотокСВебКамеры().forEach((изображение) ⇒ {
  установитьКоординатыВзгляда(
    сотворитьМагию(
      найтиГлаза(изображение, найтиЛицо(изображение)).map((координатыГлаза) ⇒ {
        return найтиЗрачок(изображение, координатыГлаза)
      })
    )
  )
})
```



Локализация лица и глаз

Стартовый набор OpenCV-джентльмена

Задача	**Решение**
Подготовка изображения	Оттенки серого
	Размытие по Гауссу
	Binary thresholding
	Оператор Кэнни
Локализация лица и глаз	Каскады Хаара
	Метод Виолы - Джонса
Локализация зрачков и радужек	Blob detection
	Преобразование Хафа

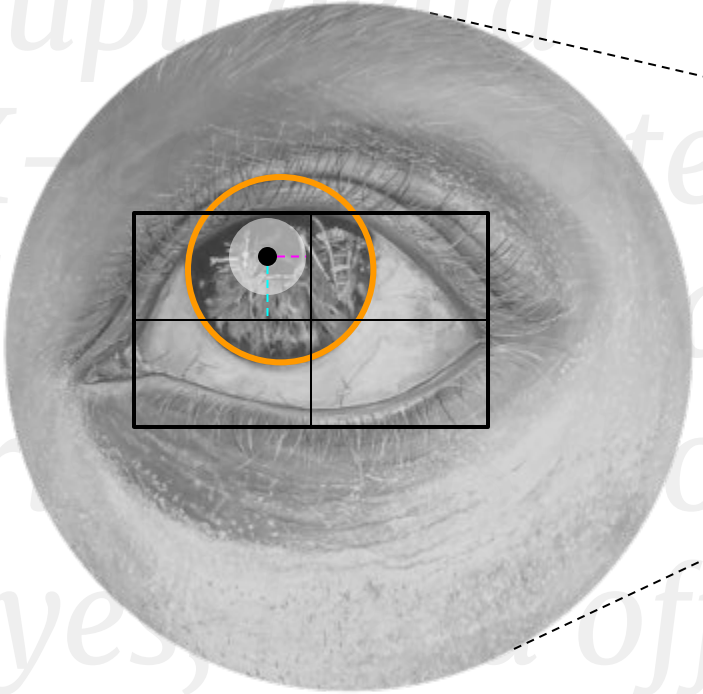


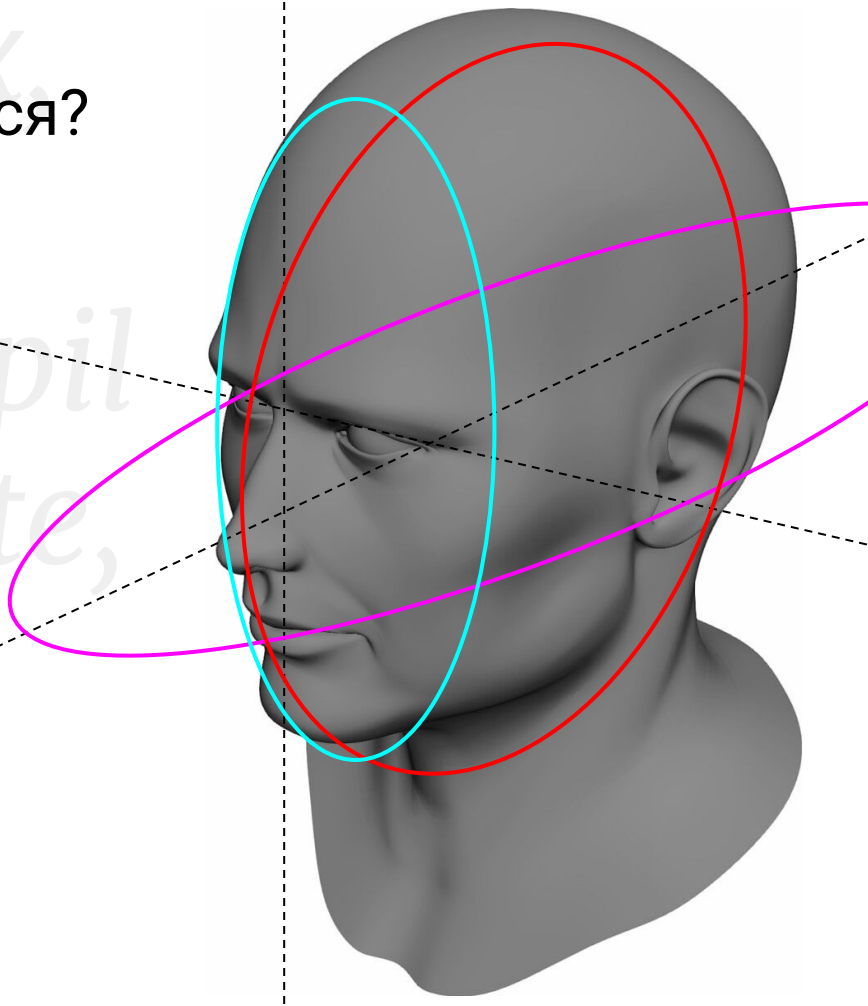
Выкидываем все перечисленное в мусорку

Без жалости, сожаления и страха

Основная задача аналитика — объяснить бизнесу,
почему построенная модель не работает.

Какие параметры нам пригодятся?



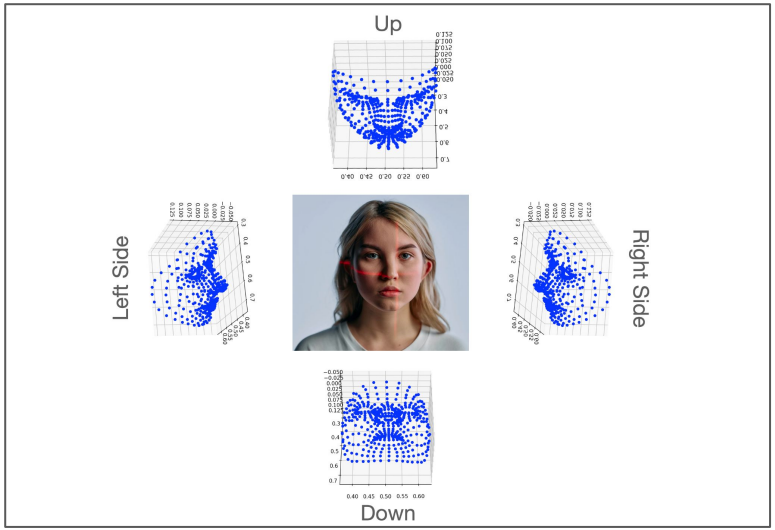


Новый план

Facial landmarks detection — Технологии цифровой лицевой антропометрии

clmtrackr, face-api.js

68 point models

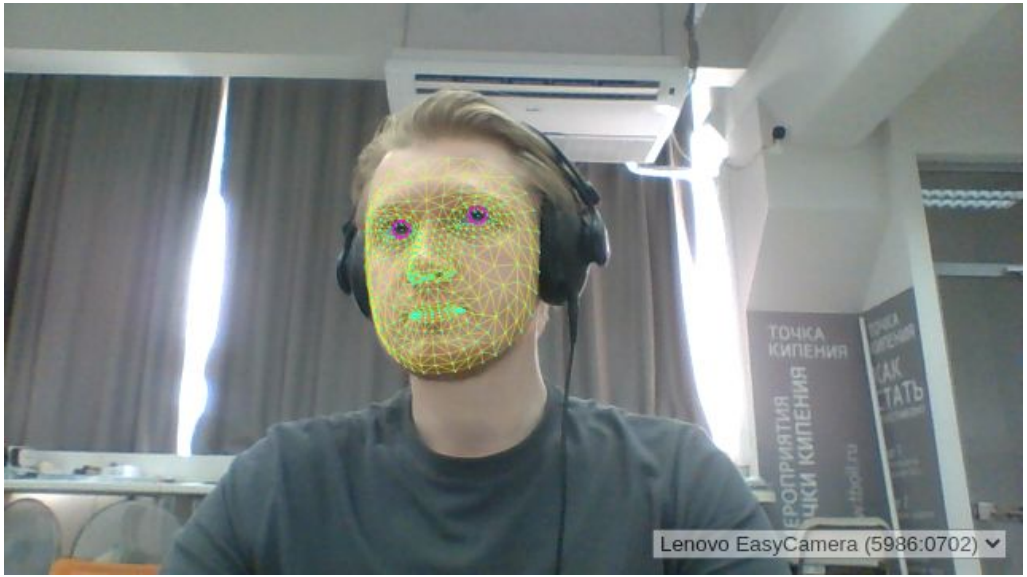


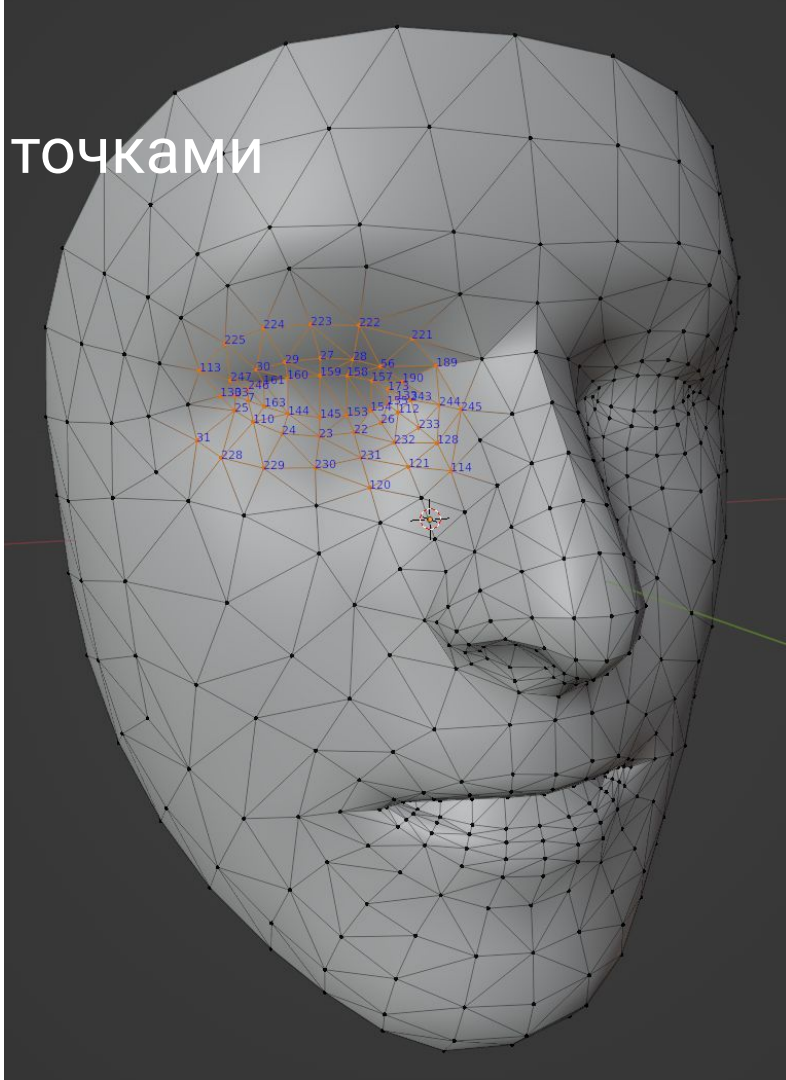
MediaPipe*, Tensorflow.js

468 point **3D** models

* доступно для Python, а еще есть пакет dlib

Полный контроль над ключевыми точками



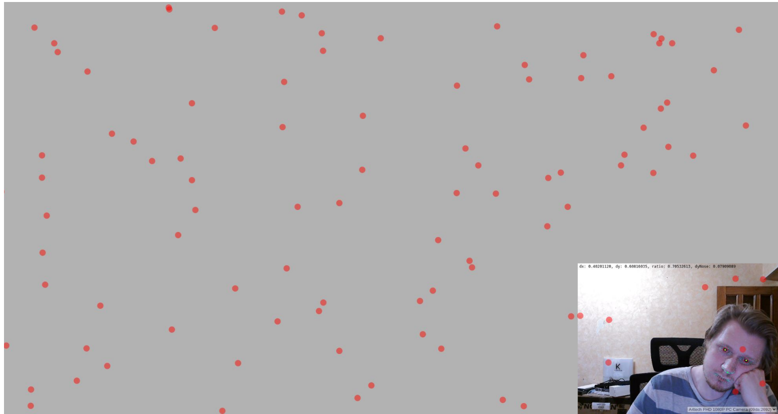


Алгоритм. Перенос координат зрачков в систему координат экрана

Сбор данных, определение корреляции

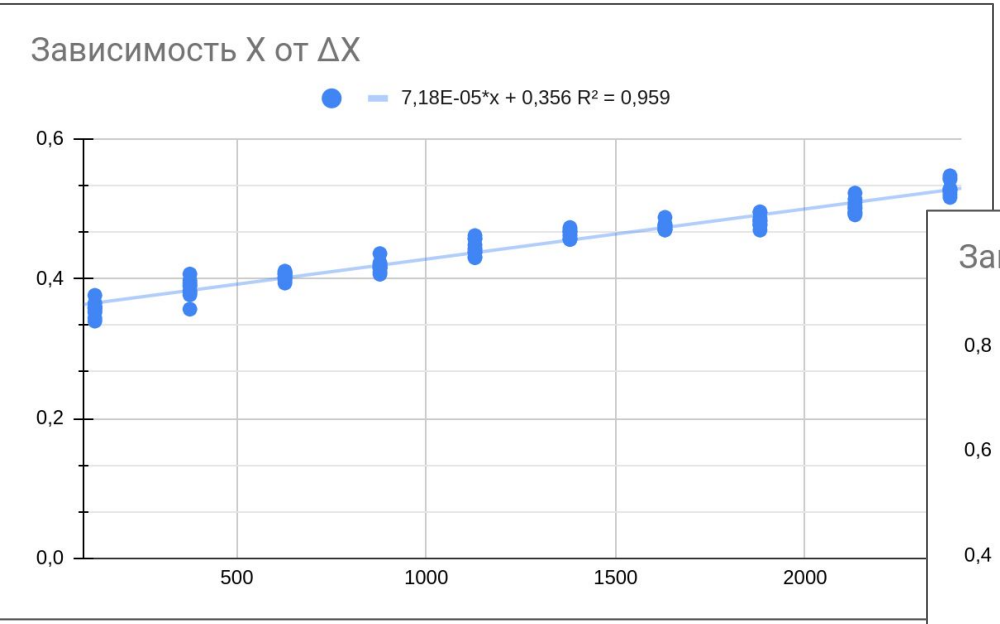
x	y	dx	dy	ratio	dyNose
125,4	67	0,3647268501	0,6520488269	0,4076740313	0,1292064786
376,2	67	0,4065406295	0,629397905	0,4226540825	0,1319127083
627	67	0,4092823743	0,6324828984	0,4355206844	0,1313761473
877,8	67	0,4187465616	0,6032893419	0,4568100533	0,1300417781
1128,6	67	0,4434410191	0,6720359179	0,4339140654	0,131177187
1379,4	67	0,4730894265	0,6362455531	0,4208501961	0,1300846934
1630,2	67	0,4737442326	0,6295152656	0,4156771296	0,1317524314
1881	67	0,4945947525	0,6242052001	0,4482072109	0,1321900487
2131,8	67	0,5223222921	0,6204722094	0,4110221228	0,1294717193
2382,6	67	0,5419814717	0,5977872325	0,4029647095	0,128088057
125,4	201	0,3515702444	0,6474512904	0,422324213	0,1284032464
376,2	201	0,3937960743	0,6508659401	0,4029947288	0,1295050383
627	201	0,4021713895	0,6229227875	0,4186853272	0,1301390529
877,8	201	0,4062471134	0,6309960296	0,4478390335	0,1304513812
1128,6	201	0,4570831229	0,6878696527	0,415650792	0,129019618
1379,4	201	0,4676171108	0,6418463202	0,41136308	0,1287472248
1630,2	201	0,4694833948	0,64289586	0,4382449262	0,1298155785
1881	201	0,4868733872	0,611468882	0,4360785897	0,127982378
2131,8	201	[illegible]	[illegible]	[illegible]	[illegible]

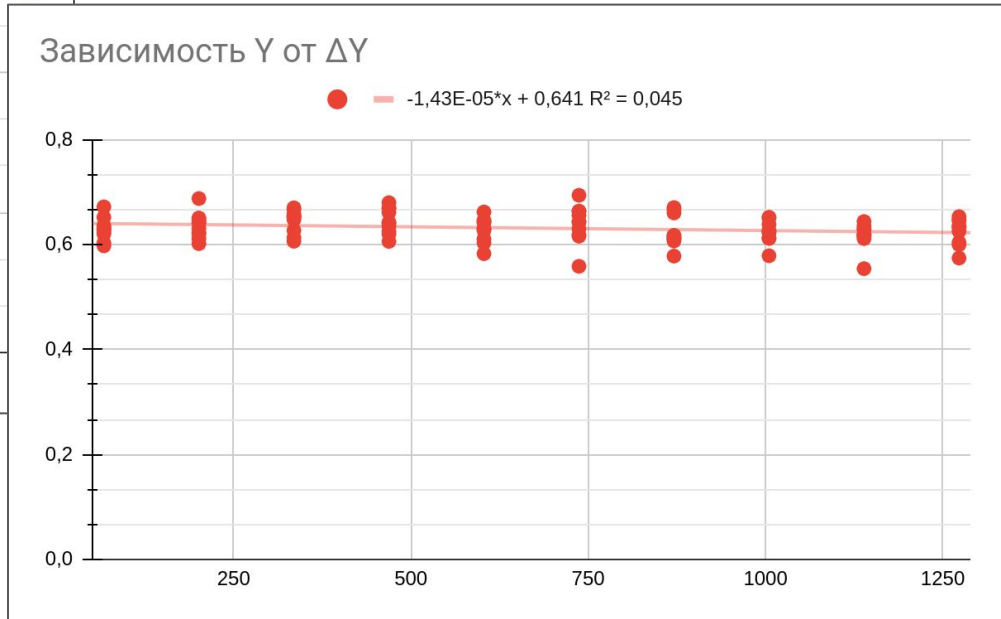
x, dx	0,9793623832
y, dy	-0,2113617221
y, ratio	-0,8260206606
y, dyNose	-0,8283451825



Алгоритм. Перенос координат зрачков в систему координат экрана

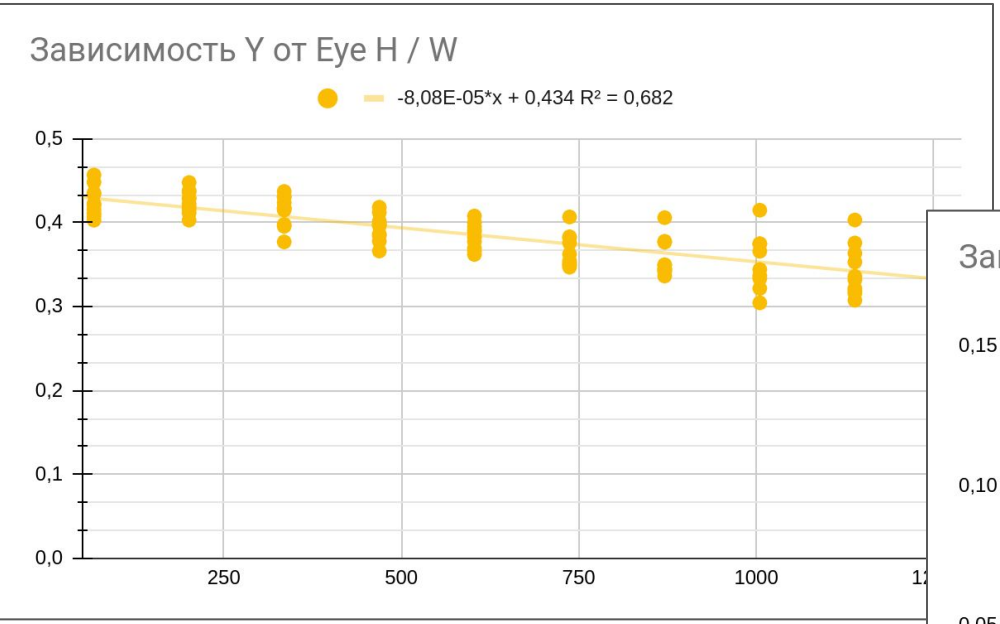
Регрессионный анализ

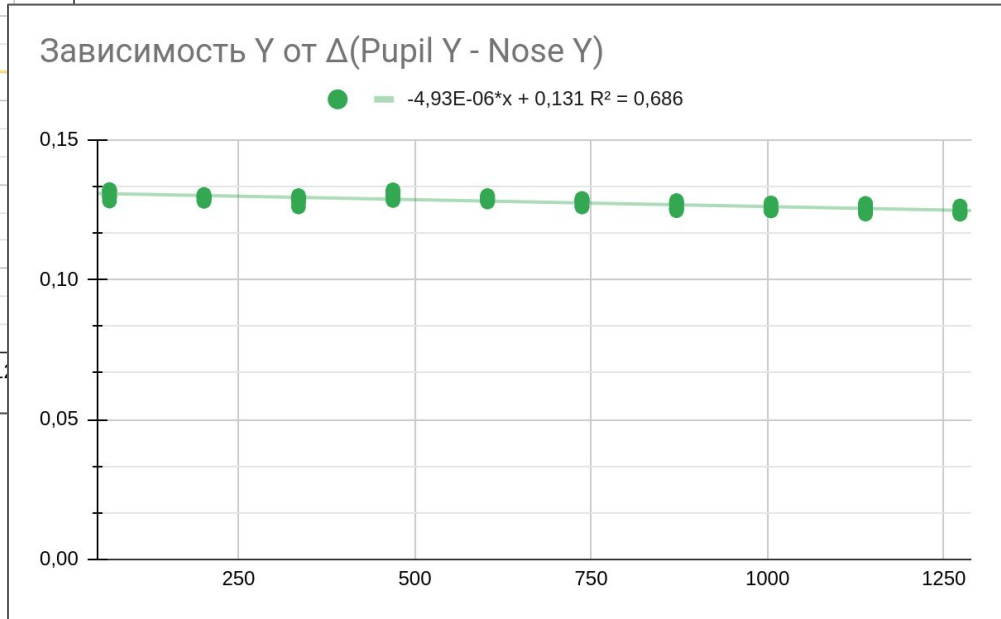




Алгоритм. Перенос координат зрачков в систему координат экрана

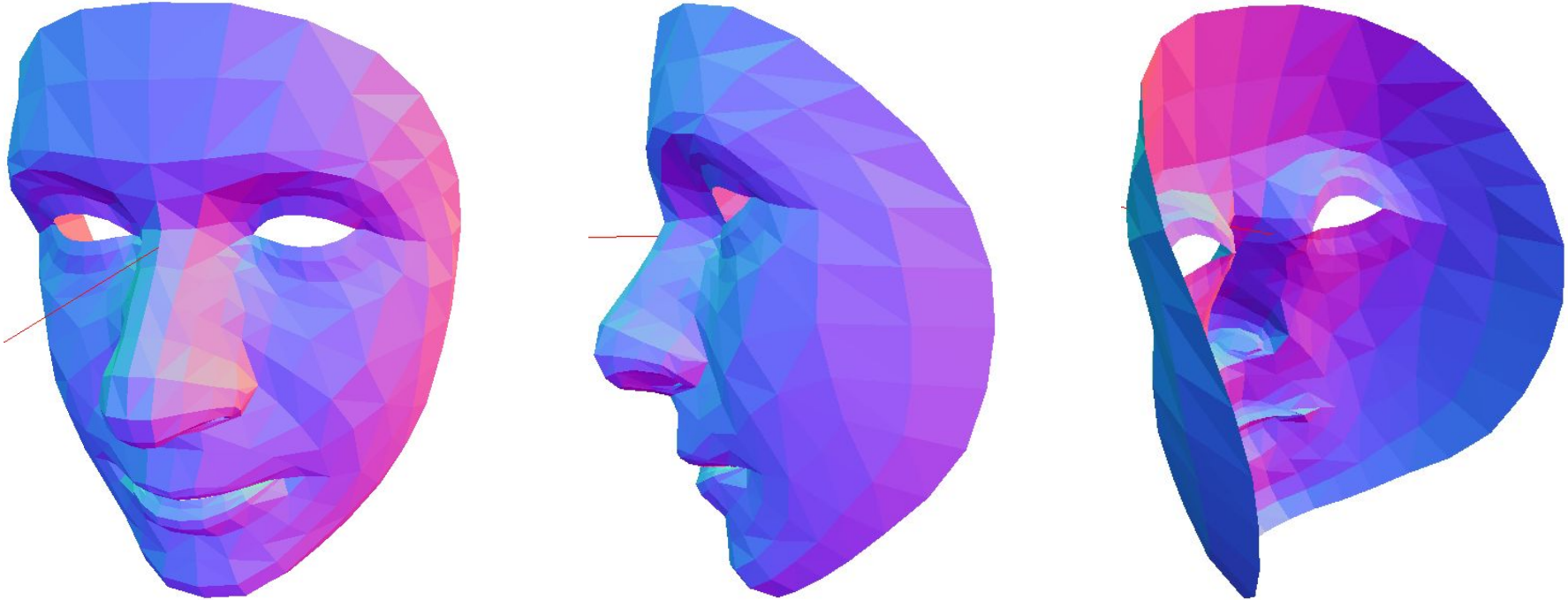
Регрессионный анализ





Определение координат взгляда на экране. Альтернативный подход

Пользуемся дарами трехмерной модели

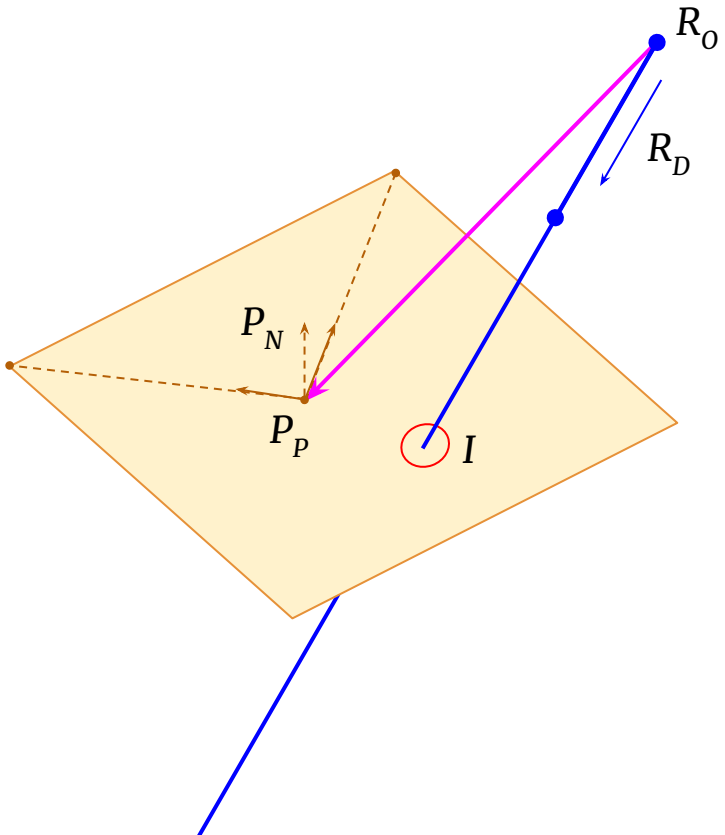


Точка пересечения луча с плоскостью

$$d_1 = (P_P - R_O) \cdot P_N$$

$$d_2 = R_D \cdot P_N$$

$$I = R_O + R_D \cdot (d_1 / d_2)$$

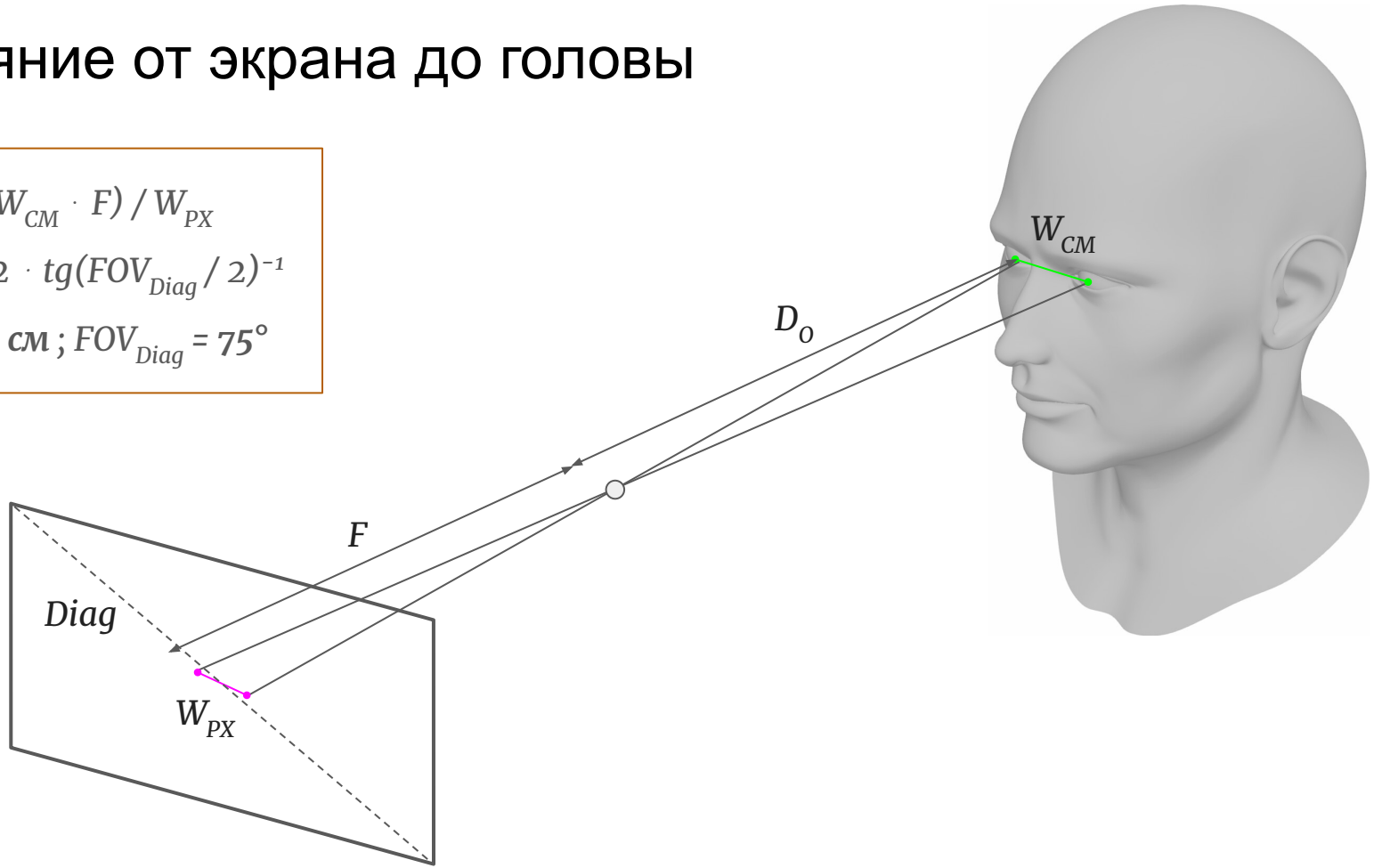


Расстояние от экрана до головы

$$D_O = (W_{CM} \cdot F) / W_{PX}$$

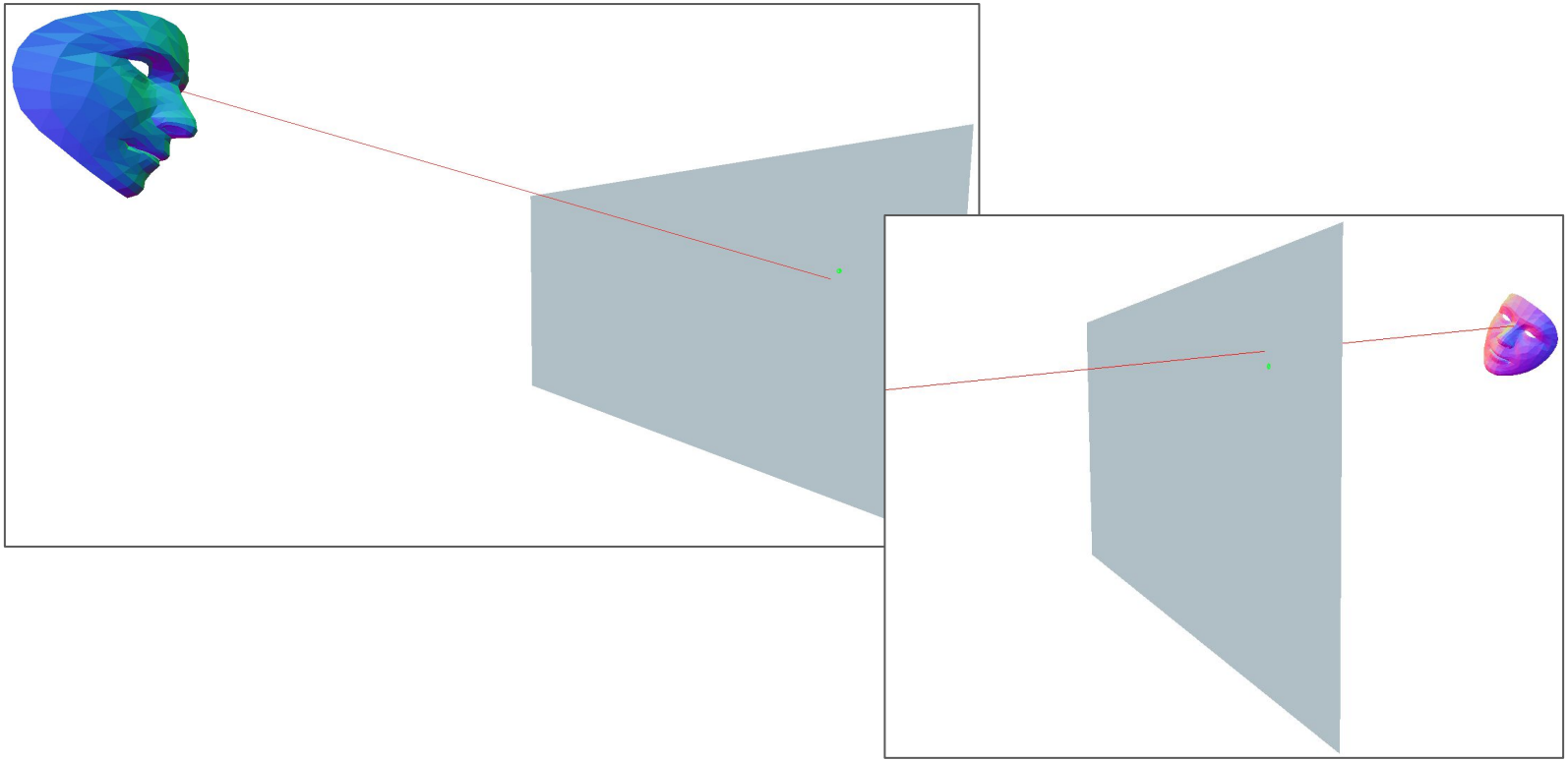
$$F = Diag / 2 \cdot tg(FOV_{Diag} / 2)^{-1}$$

$$W_{CM} = 6.2\ см\ ;\ FOV_{Diag} = 75°$$



Определение координат взгляда на экране. Альтернативный подход

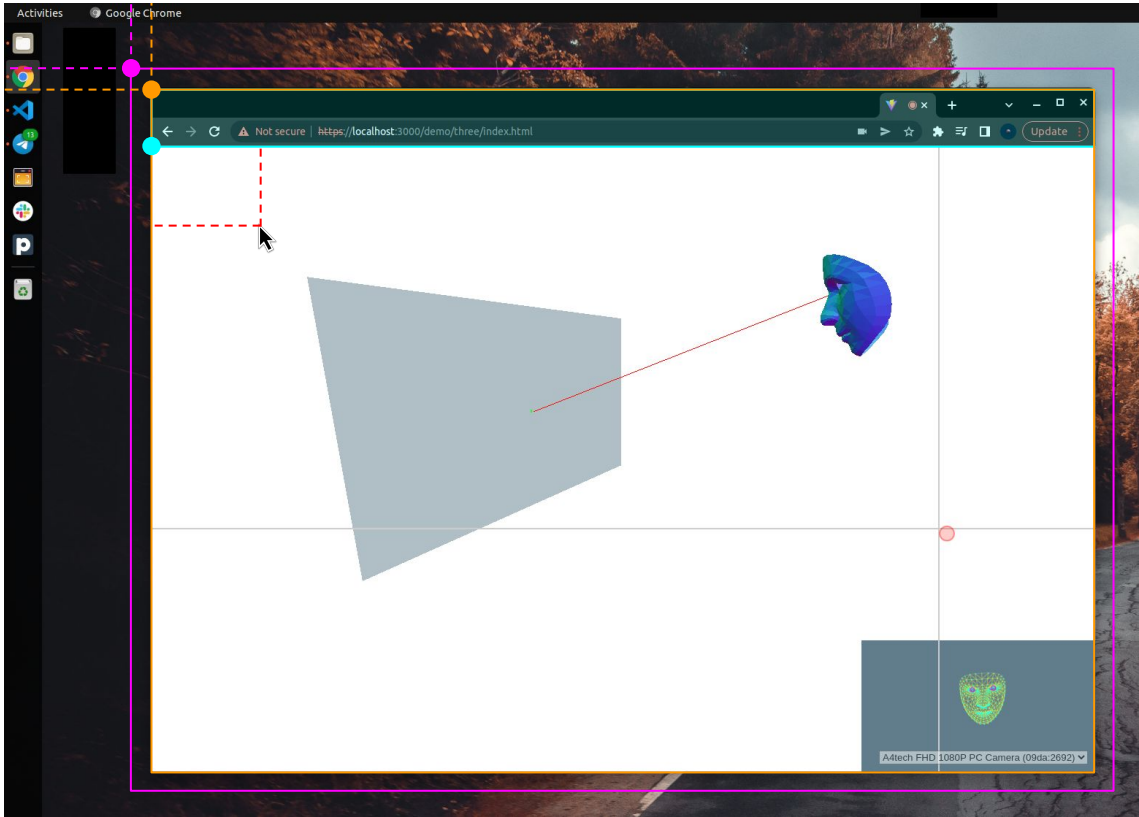
Точка пересечения вектора направления головы с экраном



Определение координат взгляда на экране. Альтернативный подход

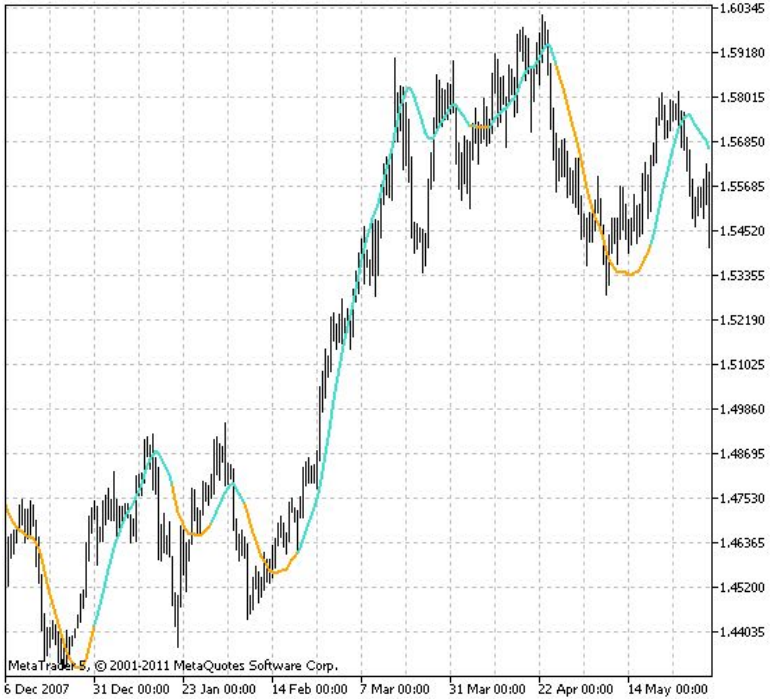
Координаты взгляда относительно центра экрана

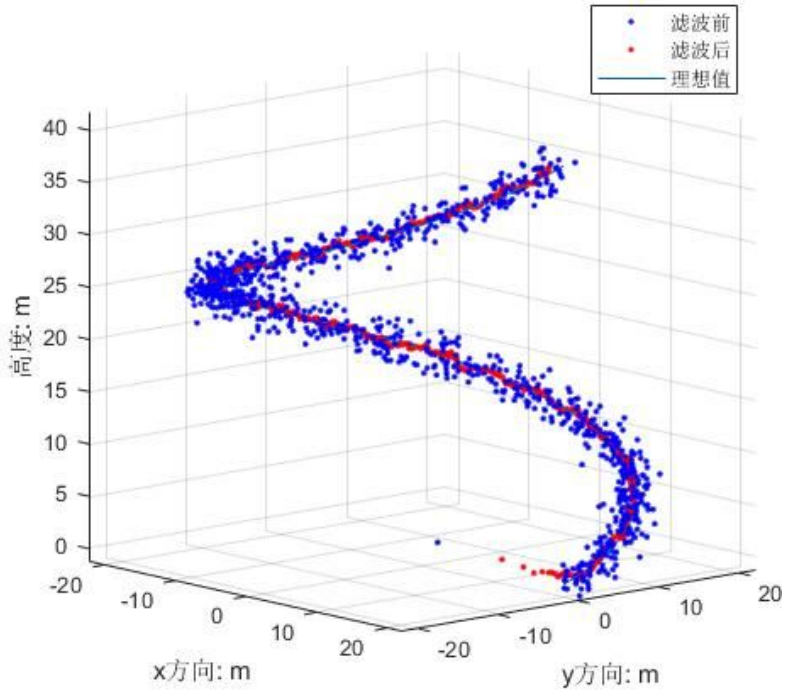
```
const offset = {
  left: 0,
  top: 0
}

window.addEventListener('mousemove', (e) => {
  offset.left = window.screenLeft - e.screenX - e.clientX
  offset.top = window.screenTop - e.screenY - e.clientY
})

const getScreenCenterRelativePosition = () => {
  const windowX = window.screenLeft + offset.left
  const windowY = window.screenTop + offset.top

  const screenX =
    Math.floor(windowX / window.screen.width) * window.screen.width
  const screenY =
    Math.floor(windowY / window.screen.height) * window.screen.height

  return new Vector3(
    screenX + window.screen.width / 2 - windowX,
    screenY + window.screen.height / 2 - windowY,
    0
  )
}
```



Определение координат взгляда на экране. Альтернативный подход

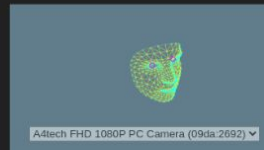
Сглаживание, фильтр Калмана





Итоги

Какие успехи и демо-версия



Внимание!!!

Спасибо за внимание