LiveVideoStackCon

聚音视　研修不止于形

2017年10月20日-21日

北京.丽亭华苑酒店

真正隐世的高手程序员

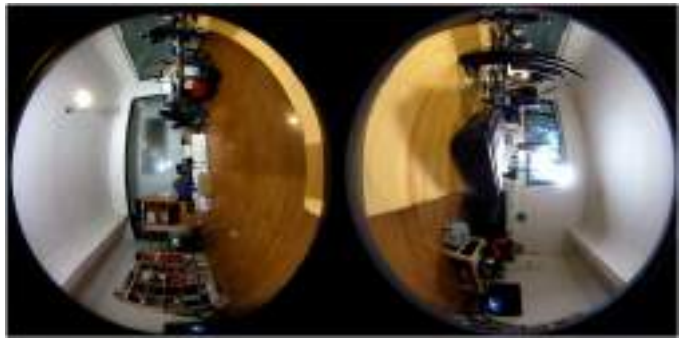
睿智，恬淡，懂得生活

他养生的秘诀是每天吃何首乌、枸杞、核桃，所以你看他现在快四十岁了精神还是很好。

自我介绍

- 我叫鲍金龙，今天45.83岁了。平时喝咖啡，吃披萨，还有健身，力量举爱好者，平衡卧推160公斤，史密斯全蹲180公斤。

应用场景介绍



HTTP TS CDN

今天要解决的问题

- 2兆码率是普通家用WIFI的经验值上限。
- 3兆码率是CDN加速和P2P传输的经验值上限。

场景	分辨率，帧速率	码率（兆）
普通电视剧	1280x720， 30fps	0.8
体育比赛	1280x720， 30fps	2.0
VR体育比赛	1920x1080， 30fps	4.0
180度VR体育比赛	3840x1080， 30fps	8.0
180度VR体育比赛	5120x1080， 30fps	12.0
180度VR体育比赛	3840x1080， 60fps	16.0
180度VR体育比赛	5120x1080， 60fps	24.0

普通电视剧

- 1280x720， 30fps， 800kbps 效果不错了。

小羊肖恩，轻松愉快。

体育比赛，歌舞表演，文艺晚会

- 1280x720， 2兆打底。

小羊吃了膨大剂，大胖羊了。

体育比赛VR

- 1920x1080， 30fps, 4兆码率杠杠滴。

坏了菜了，一左一右两只大胖羊

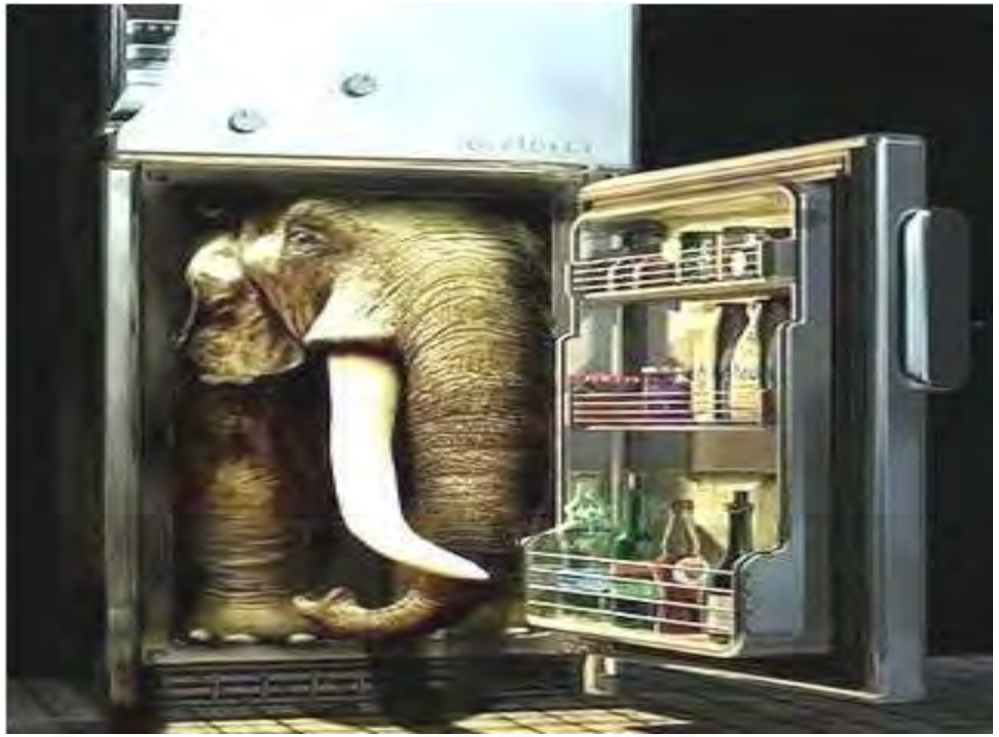
180度全景VR

- 3840x1080，30fps。码率：估计得8兆以上。这效果还不算好，5120x1080才比较好。30fps不太好，50，60fps才好。那码率得多少啊？不可想象。

两只大肥羊都不行了，大象来了。

问题来了

- 10兆以上的码率和3兆的上限，就如同要把大象塞进冰箱，怎么办才能把大象塞进冰箱去呢？



今天分享什么东西

- 1）不仅仅局限于这一个案例。
- 2）主要介绍经过实战检验的思路，方法。
- 3）带有创新点的内容。
- 4）算法优化。
- 5）不涉及编程知识/技巧和工具软件的使用。

关键字

- 1）实用主义
- 2）两面墙
- 3）两只眼
- 4）三个梯队

实用主义

- 电影阿波罗13号，美国专家的表现

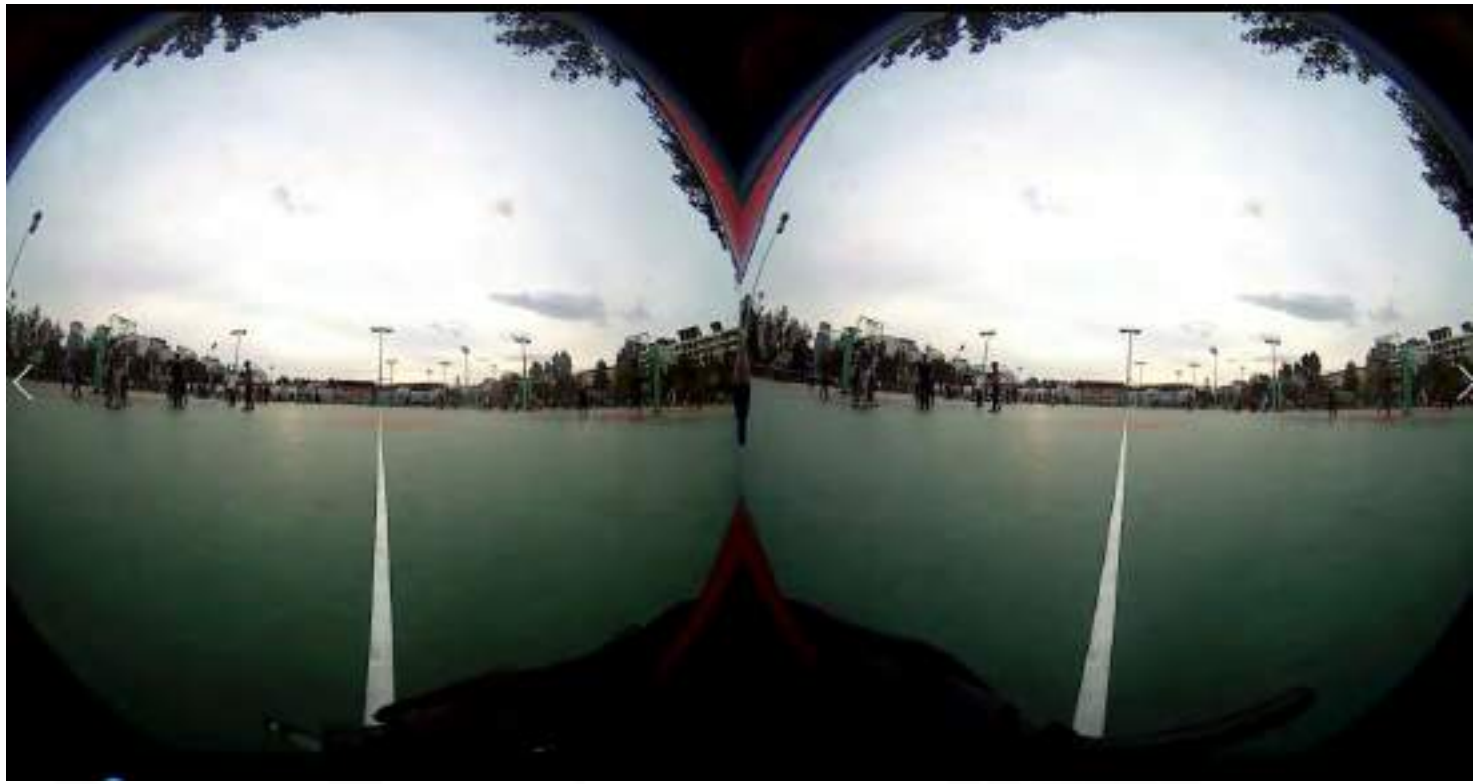
第一个视频-
APOLLO13

危急时刻，中国老专家逞威

东北一个化工厂，也是油罐着火

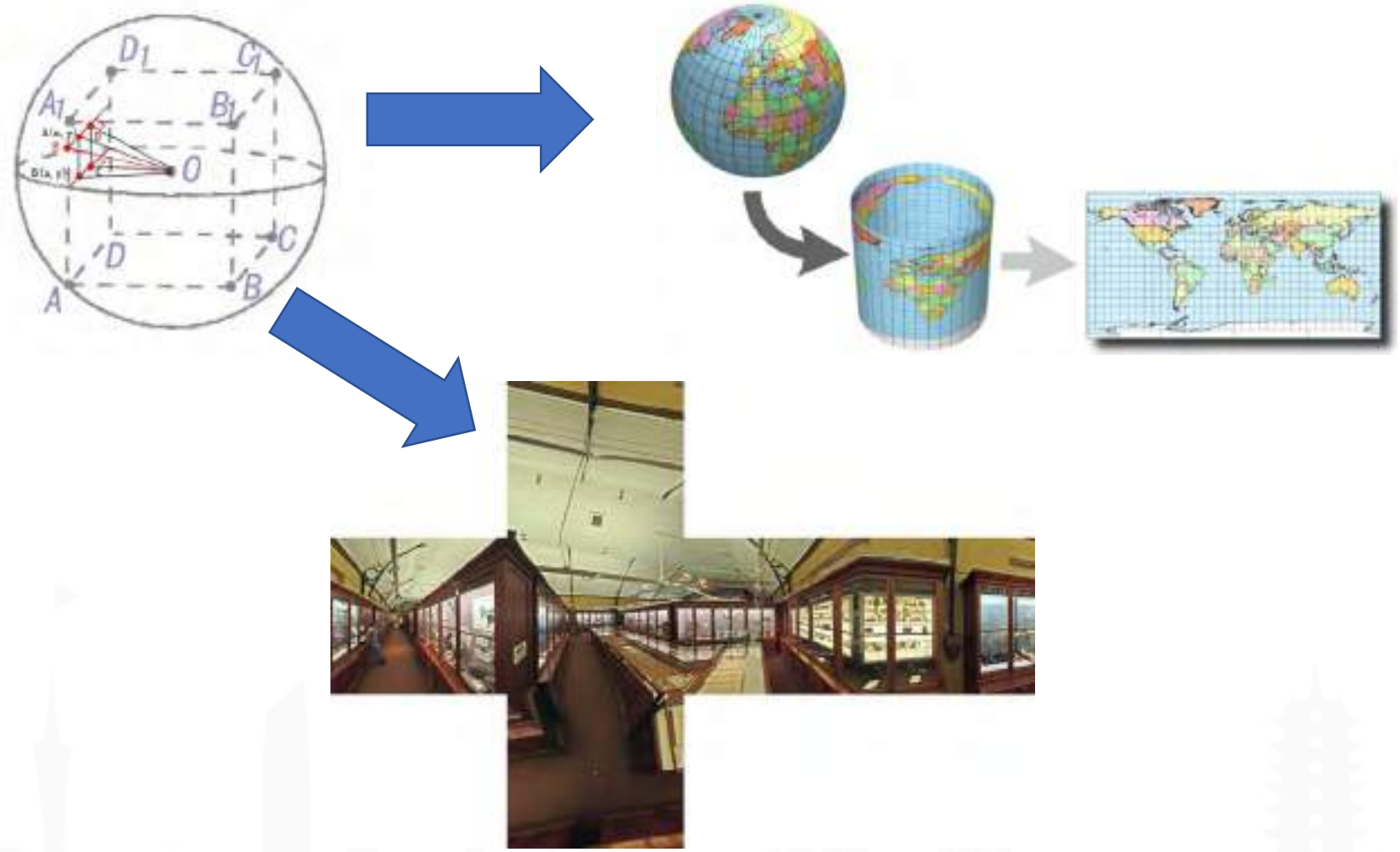
优化第一步，两面墙

- 效率问题，太多的蓝天白云

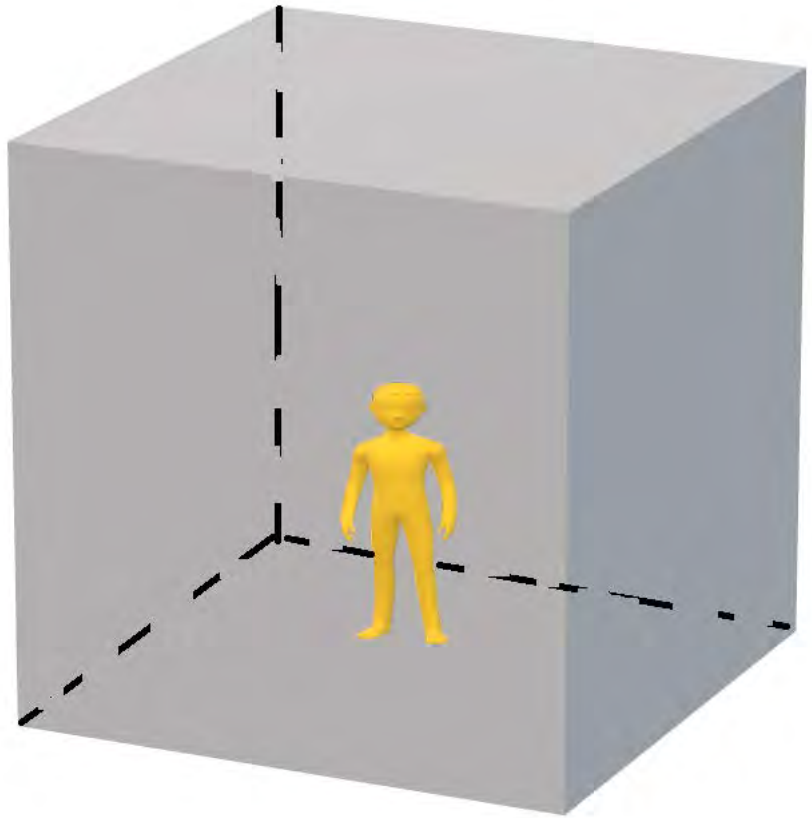


第二个视频-
360VR

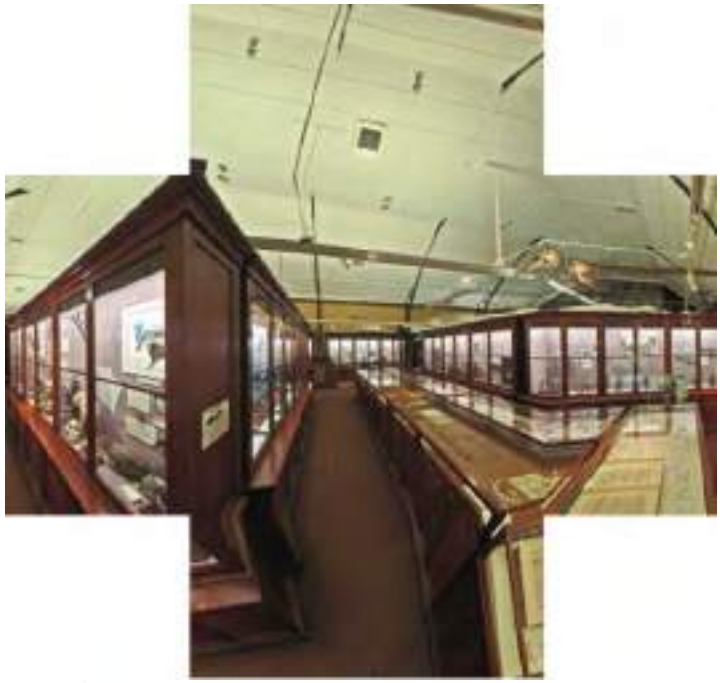
从映射做起



全封闭透明包厢

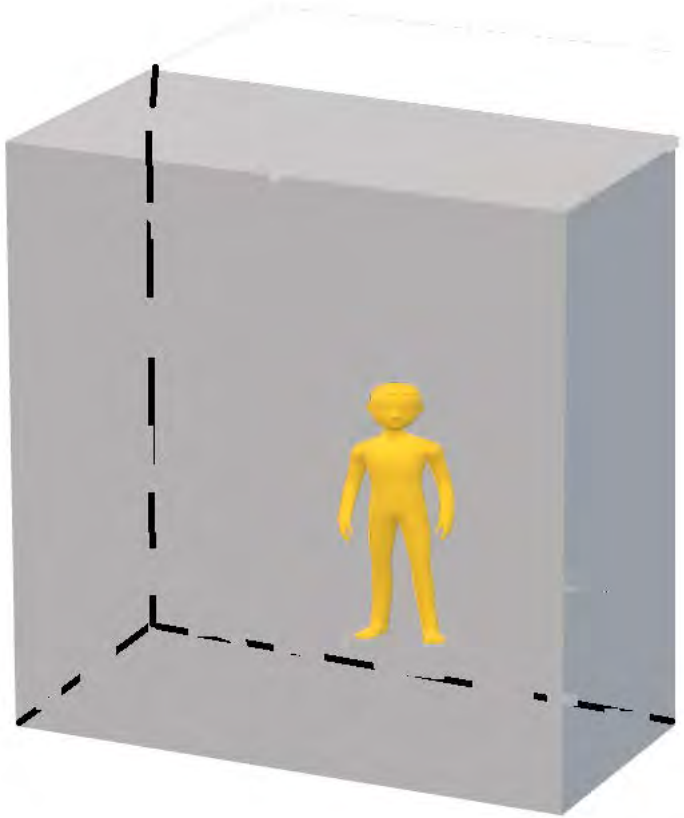


180度全景，场景去掉一半

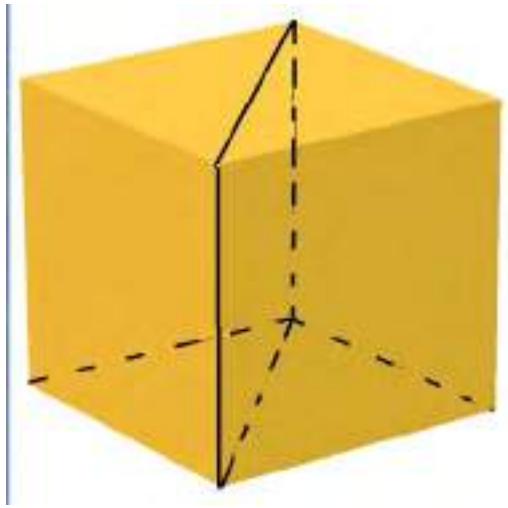


去掉盖子

180度包厢

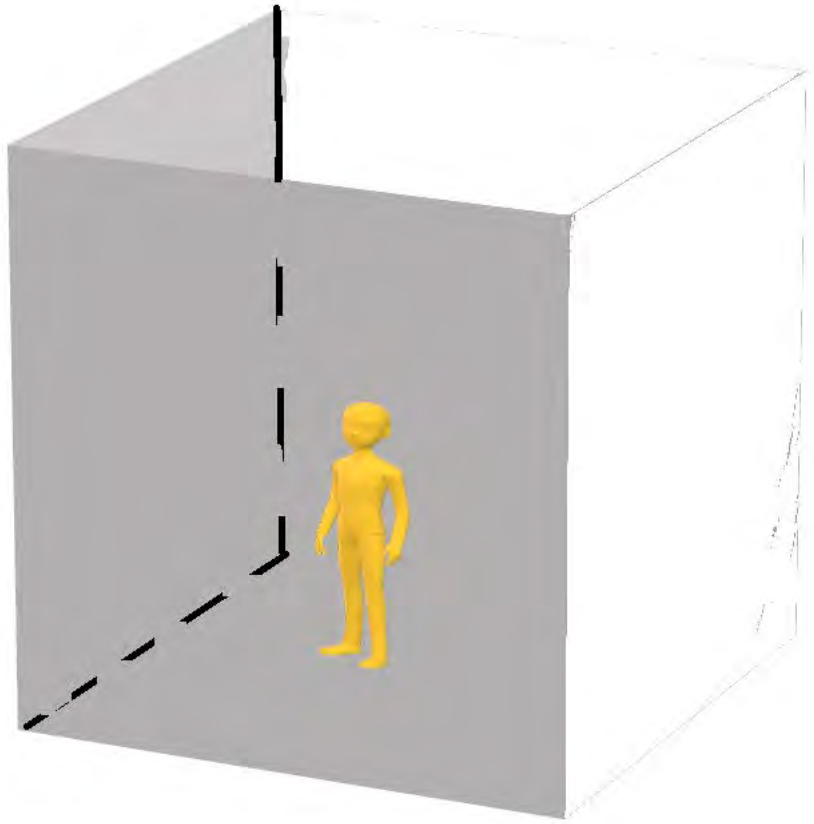


更简单的切法

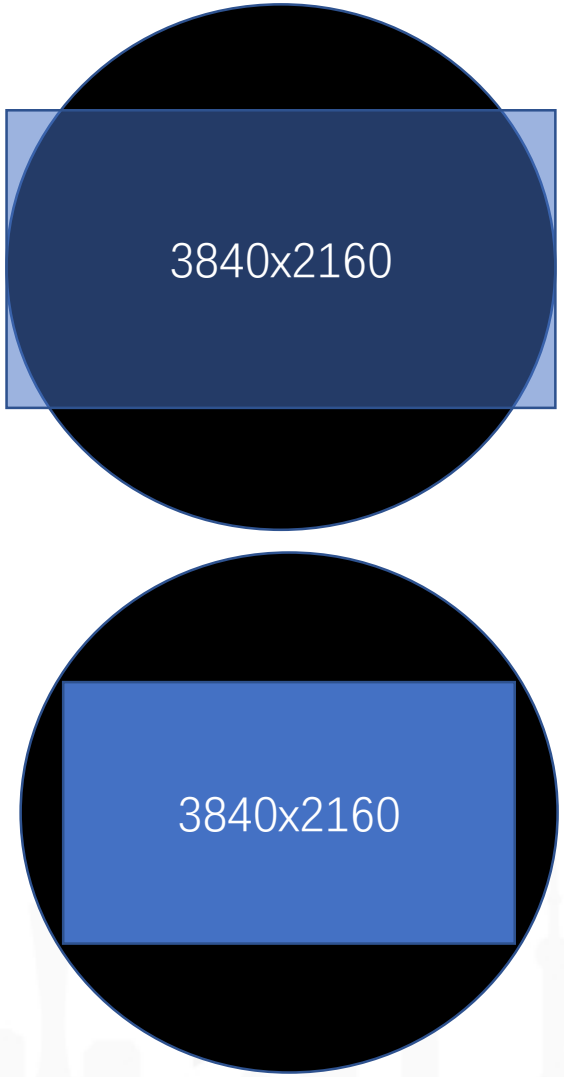


取两面

青春版的包厢

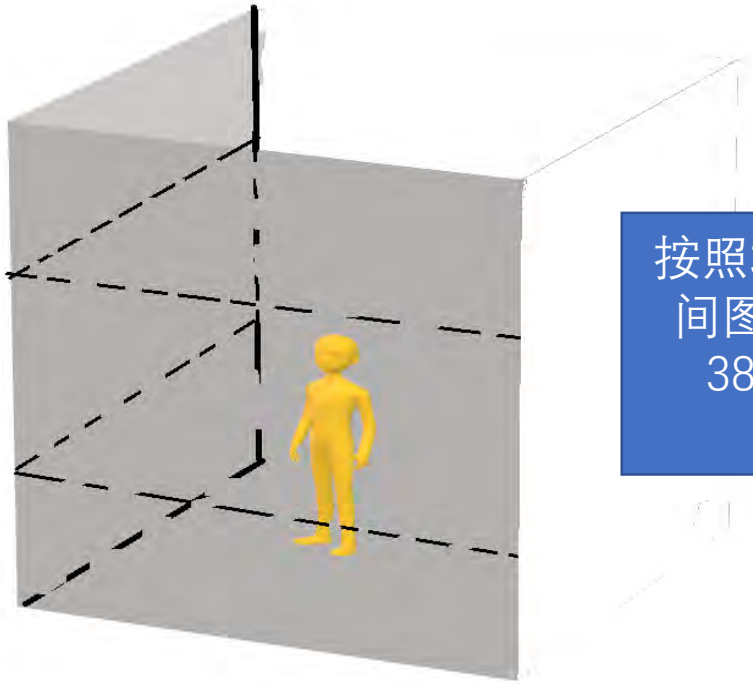


提高CCD使用效率



贴到球面

超宽幅环绕银幕的效果



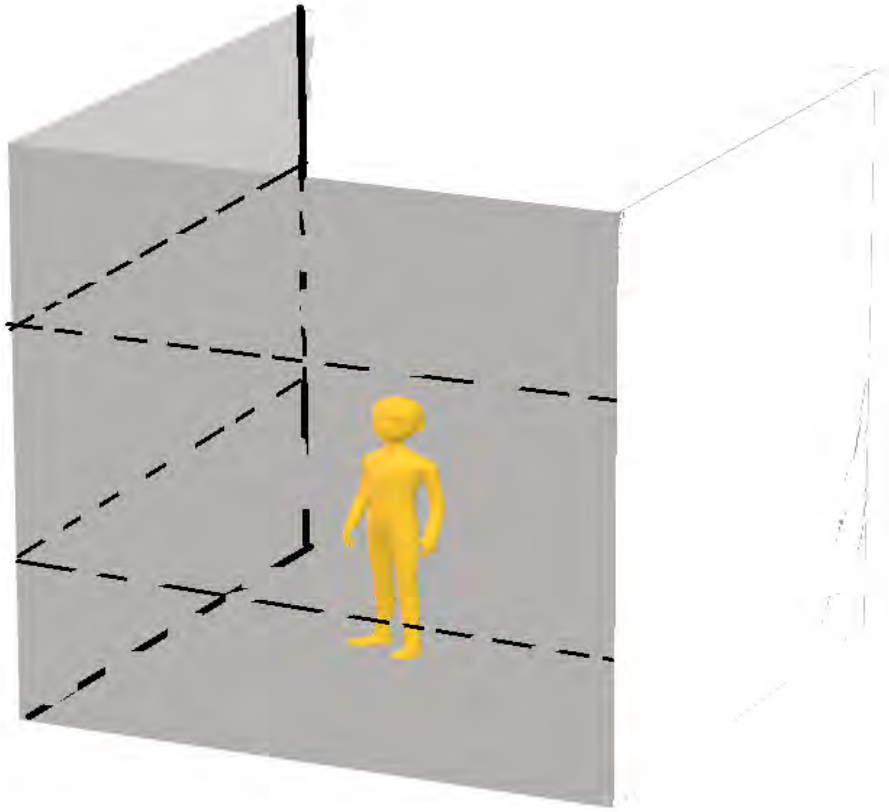
按照32：9 取中间图象保存，3840x1080

提高像素利用率

水平压缩一半，LR排列32：9

CUBE两面墙映射的好处

两面墙，超广角采样，内容的减少使码率大概降低了15%。因为映射效率的提高，可以使用更低的分辨率，使用　　2560x960 替换3840x1080，实际上使得码率降低了一半。



下面该HEVC出场了

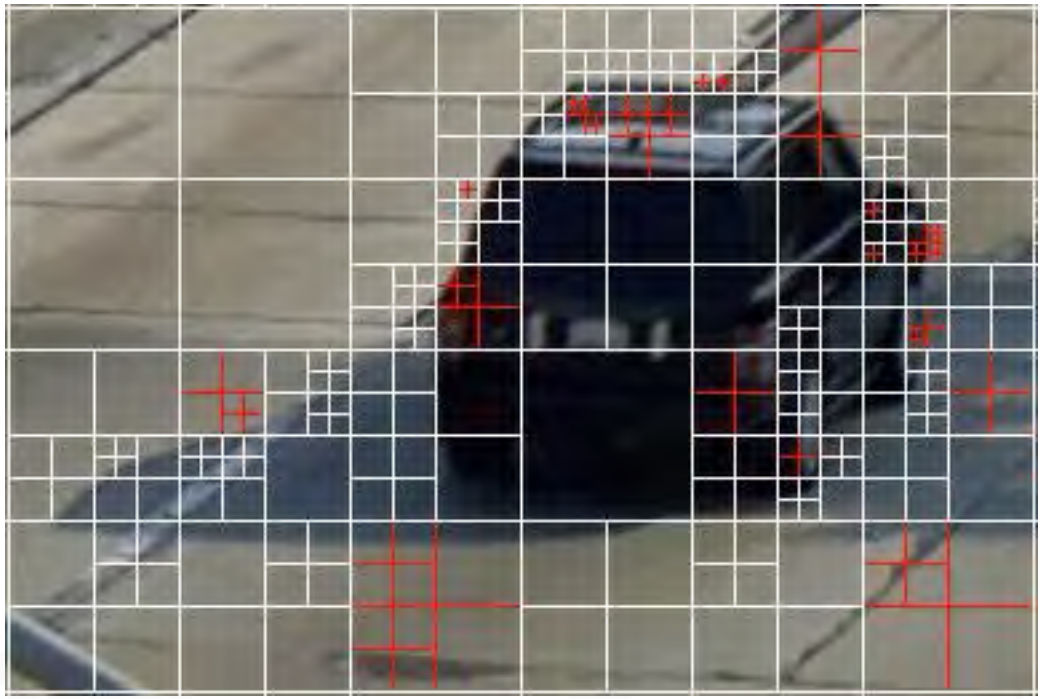
1）编码标准：HEVC
2） 编码工具

X265出场了

- x265为什么码率低，质量高
  - 参数可调配
  - 支持动态量化

动态量化

X265在每一帧里面，量化参数QP是可以跟着CTU动态调整的。



什么是两只眼

Cube map的180度VR
HEVC-3D?
MVC?
NEXTVR?

FS排列

FS为什么好一点呢？
按道理来说，MVC的intra pred比较合理，因为两个视角的图像不是完全匹配的，不光是水平位移，还有角度变化。但事实上FS要好一点，可能是因为符合inter pred的情况比较多。

“两只眼”降低了30%的码率。

分辨率	CRF	FPS	LR排列码率	MVC码率	FS码率
3840x1080	28	30	8兆	6兆	
1920x1080	28	60			5.4兆

三个梯队

但是5.5兆跟3兆的目标还有不少差距的呀。还得想辙。

划分的依据

- 你是来看比赛的，不是来看观众的。所以篮球场这一块清晰了就行了。观众那就都当糙哥去了。
- 那怎么分三部分呢？你看比赛看什么？看拿球的那个，还有附近防守的人。所以你就在篮球附近画一个圈子就行了，这里面的好好对待，球场上的差一点，观众呢还是糙哥。
- 有人问了，那要是练球时候有多个球你追踪哪个？太简单了，靠近篮框的那个就行了。合理吧？
- 小明同学就说了，要是在中线有俩球来回震荡呢，刷刷刷，是不是编码器就疯了，不知道怎么办了？

小明同学请你粗去

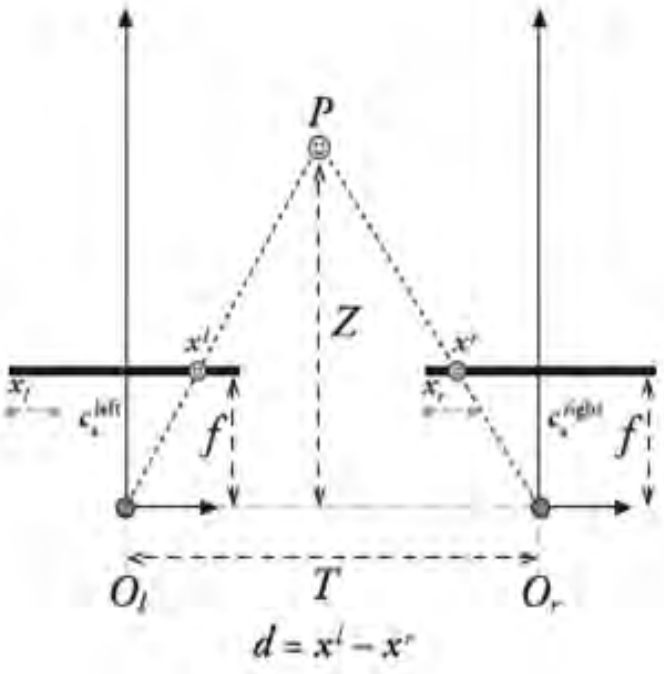
YY一下能挣多少

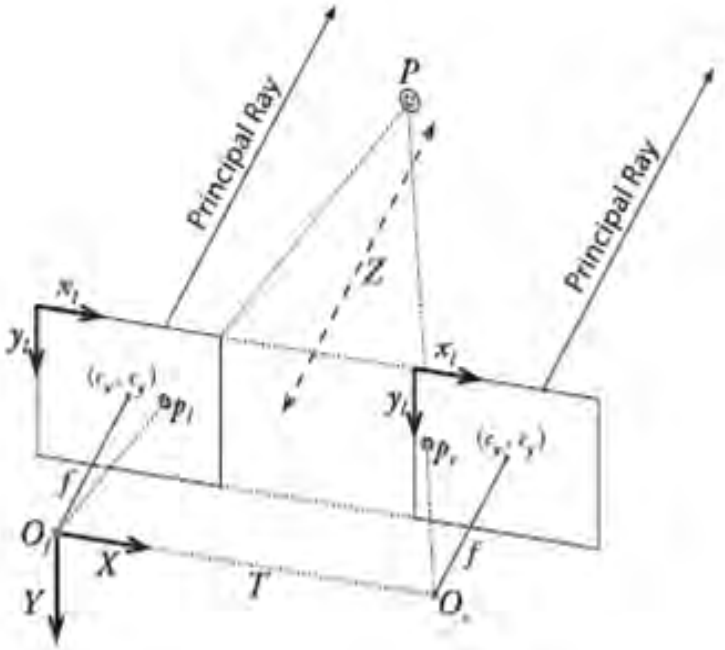
- A, 面积占50%， 1兆码率。
- B, 面积40%，2兆码率。
- C, 面积10%，6兆码率。这样的话码率是多少呢？
- 1x0.5 + 0.4x2 + 6x0.1 = 1.9 ！ 哇超额完成任务了。
- 1x0.5 + 0.4x3 + 6x0.1 = 2.3 ！ 哇超额完成任务了。
- 1x0.5 + 0.4x4 + 6x0.1 = 2.7 ！ 哇超额完成任务了。
- 那么简单一点，分两部分，3兆码率的话，两部分怎么分合适？
- 1x0.5 + 5x0.5 = 3，效果稍差一点，马马虎虎了。

技术需求

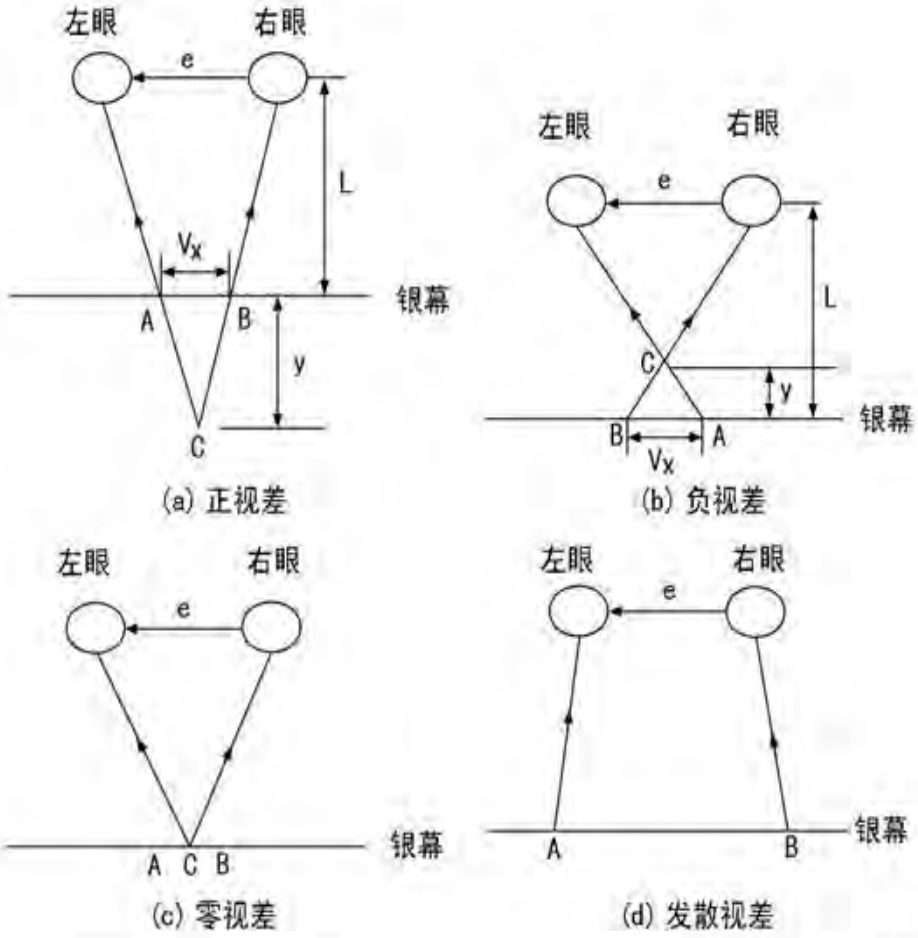
- 第一个，把篮球场切出来。这个只要有深度信息和CTU的对应关系就行了。恰恰咱们做的是双摄像头图像，就擅长干这个。
- 第二个就不容易了，你看首先你得认识篮球，还得认识篮筐，可能还得认识防守和掩护的队员。所以这一定得经过物体识别和跟踪的过程。
- 第一个方案其实就马马虎虎够用了，所以咱们先做简单的。

两种聚焦模型-平行聚焦





双目聚焦



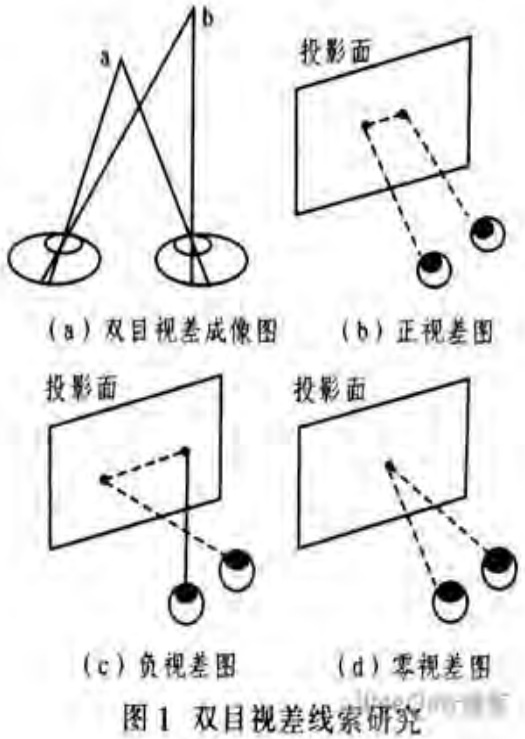
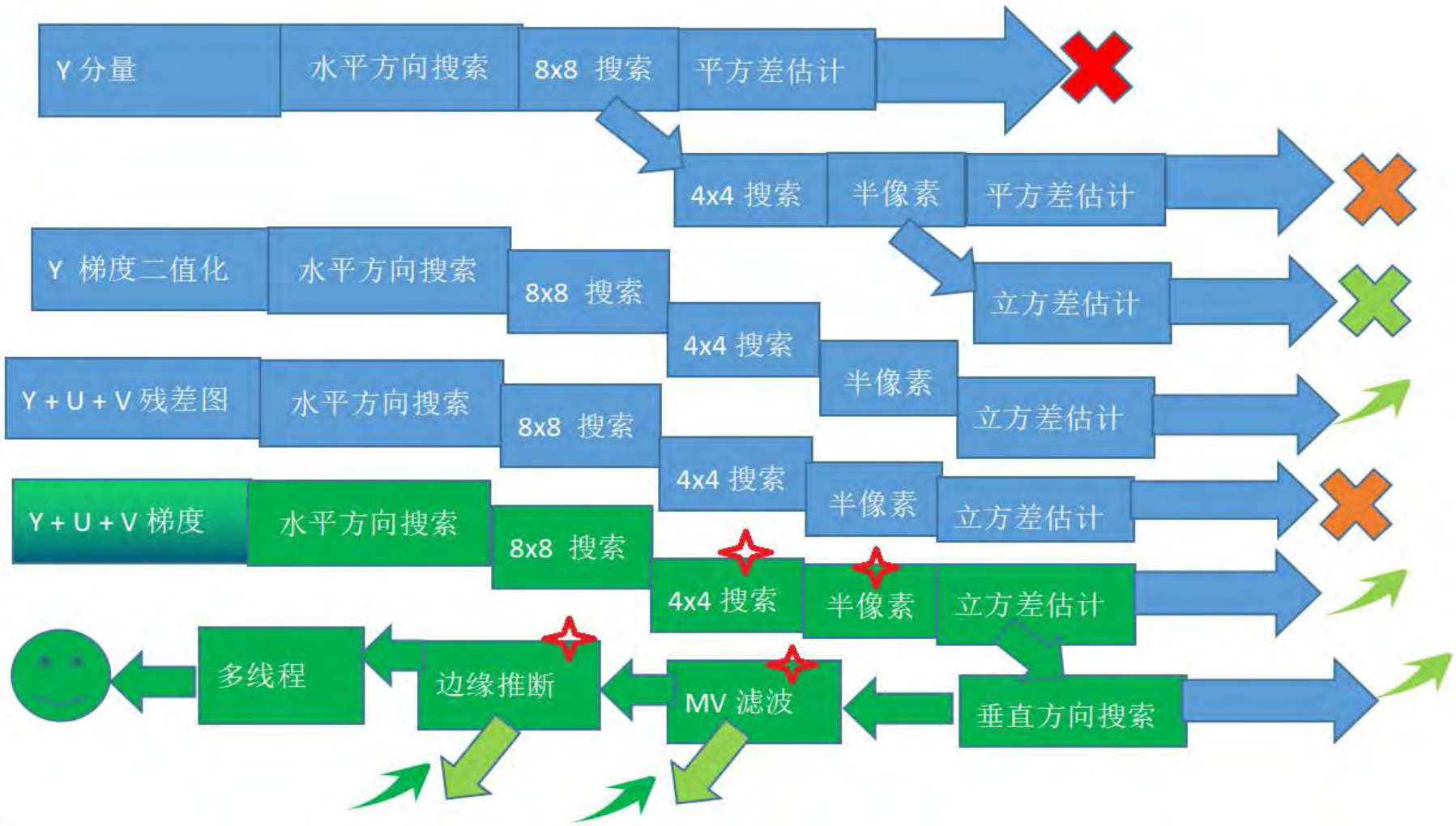


图 1 双目视差线索研究

艰苦卓绝的开发过程


Y 分量
水平方向搜索
8x8 搜索
平方差估计
4x4 搜索
半像素
平方差估计
Y 梯度二值化
水平方向搜索
8x8 搜索
4x4 搜索
半像素
立方差估计
Y + U + V 残差图
水平方向搜索
8x8 搜索
4x4 搜索
半像素
立方差估计
Y + U + V 梯度
水平方向搜索
8x8 搜索
4x4 搜索
半像素
立方差估计
多线程
边缘推断
MV 滤波
垂直方向搜索

物体追踪过程

第三个-object-detect

梯度图

第四个视频
-gradian

深度信息图

第五个视频
-depth

性能优化

- 最后的版本，处理3840x1080的LR图像，在intel i7 6820hk 上面大概需要30ms。这就是说如果是这个分辨率，60fps FS排列需要4个CPU核心，100fps需要8个核心。X265在16核以上的系统中，恰好有25%左右的CPU空闲，这个正好使用剩余的CPU时间，不浪费。

最终聚焦方案

- 识别比较成功的是双目视差的焦点附近的物体。
- 而且双目视差的主观体验比较舒适。
- 因此目前我们采用双目视差来拍摄，预处理应用在图像中心部分处于焦点附近的物体。

分层处理的收获

- 深度信息有了，还有一个很重要的工作没做，那就是你使用了多个动态QP，人家X265支持吗？当然不支持了。所以要自己改代码。这个工作量还是不小的。
- 简单分层的码率降低15%左右。

三管齐下的总体收获

- 1）两面墙，超广角采样，内容的减少使码率大概降低了15%。因为映射效率的提高，可以使用更低的分辨率，使用 2560x960 替换3840x1080，实际上使得码率降低了一半。
- 2）两只眼降低了30%。
- 3）简单划分大概降低了15%。
- 从3840x1080 LR 30FPS, 到2560x960 FS 60 FPS,码率从10兆降低到了2.5兆左右。总体上降低了到原来的25%。2.5兆的码率效果可以接受。

存在的问题

- 1）近距离物体的视角差别造成物体形状和大小的不匹配。
- 2）远距离物体视差超过了物体本身尺度，相似物体无法区分。
- 3）视角的改变造成物体叠加关系的改变。
- 4）超大数量的物体造成分割和归类实际上无法完成。

X265 benchmark

16x HWBOT x265 Benchmark - 1080p排名

1	Splave Core i9 7960X	180.16 fps
2	sofos1990 Core i9 7960X	179.26 fps
3	Hicookie Core i9 7960X	168.44 fps
4	Bruno Ryzen Threadripper 1950X	89.61 fps
5	CL3P20 2x Xeon E5 2667 v4	87.84 fps
6	TheOverclocker Ryzen Threadripper 1950X	84.97 fps
7	Atlas Rush Ryzen Threadripper 1950X	84.07 fps
8	Bilko Ryzen Threadripper 1950X	75.87 fps
9	blueleader Ryzen Threadripper 1950X	73.21 fps
10	CHRIS_666 Ryzen Threadripper 1950X	72.52 fps

16x HWBOT x265 Benchmark - 1080p排名

5	CL3P20 2x Xeon E5 2667 v4	87.84 fps
6	TheOverclocker Ryzen Threadripper 1950X	84.97 fps
7	Atlas Rush Ryzen Threadripper 1950X	84.07 fps
8	Bilko Ryzen Threadripper 1950X	75.87 fps
9	blueleader Ryzen Threadripper 1950X	73.21 fps
10	CHRIS_666 Ryzen Threadripper 1950X	72.52 fps
11	ajc9988 Ryzen Threadripper 1950X	69.62 fps
12	andressergio 2x Xeon E5 2667 V3	62.26 fps
13	L0w Budg3t 2x Xeon E5 2680	48.47 fps
14	Eugene 2x Xeon E5 2670	46.16 fps

请大家关注11，这个是单cpu 16核心 ， 3.3GHZ没有超频的成绩。因为这个性价比最高。1080P 70fps左右（2倍速多点），这成绩对于60fps来说就够了。前面说过，剩下的CPU时间刚好做预处理。

X265能不能更快

- 参考帧的数量对速度的影响最大，一个参考帧最快。其次是CU的最大最小尺寸，只分32 和16两级最快。
- 左眼图像只参考左序列前一帧，这相当于ultrafast。
- 右眼图像先参考左眼图像，后参考右序列前帧。
- CU还是最大64到8。
- 这样的性能大概提升了50%，大概是110帧。
- 超过16核心用分段编码。

后续开发的关键点

- 1）增加多尺度的搜索划分块(AVX-512,HVX-1024)，力图避免相似物体的混淆。
- 2）寻找合适的滤波器来解决物体角度改变的问题。
- 3）通过分割或者卷积来解决物体重叠关系问题。
- 4）增加时间方向上的参考序列，用渐进的方式逐步完成对各个局部的物体分割，并建立在时间方向上的物体运动模式，提高物体匹配成功率。
- 5）使用GPU加速。
- 6）增加可以调节焦点的双目视差的辅助摄像机，来完成对场景的深度数据的计算。

三层划分的设想

- 刚才咱们说的这一切， 都是说的是2层的划分，简单的做一个深度探测，在球场上竖起一道虚拟的高墙，把打球的队员们隔离出来优待。
- 那么三层划分怎么做呢?
- 三层划分要动用DNN或者CNN工具了。
- 这里面的思路就是两条线索，
- 一个是建模之后的训练，这在目前看来必须是GPU才能完成的。
- 第二个是结合编码器的推断预处理，这个有两个思路，一个是GPU，另一个是使用AVX-512,HVX-1024来优化卷积运算。
- 涉及的工作量太大了，现在还没有一个明确的可用的demo出来。
- 等有结果咱们再找机会分享吧。

立体视觉和MR探秘

- 黑夜给了我绿色的眼睛，我就用它来寻找光明。
- 7亿年前的一只小虫从海藻那里借来了一个叶绿体基因，用这个基因生成了世界上第一只眼睛，一个小小的感光点。这个基因的二倍体生成了昆虫的复眼，这个基因的四倍体生成了带有晶状体和视网膜结构的摄像机类型的眼睛。
- 《Leaps.In.Evolution.Series.1.1of3.The.Origin.Of.Eyes》

果蝇幼虫的算法

- 果蝇幼虫的眼睛只有24个光受体（人眼包含的光受体超过1.25亿），从它们眼睛输入的光勉强够大脑把这些光点加工成像。这些幼虫会来回摇摆它们的头，以一种类似于扫描的方式来探测事物，如此能收集更多光点，让大脑构建出一幅活动的全景图，清晰到足以“看见”事物。摇头扫描能帮幼虫把更多视觉输入收集在一起，那些严重视力下降的人在光线昏暗时也常常来回摆动他们的头，以此采集足够的光线来形成大脑图像。

从听觉产生视觉

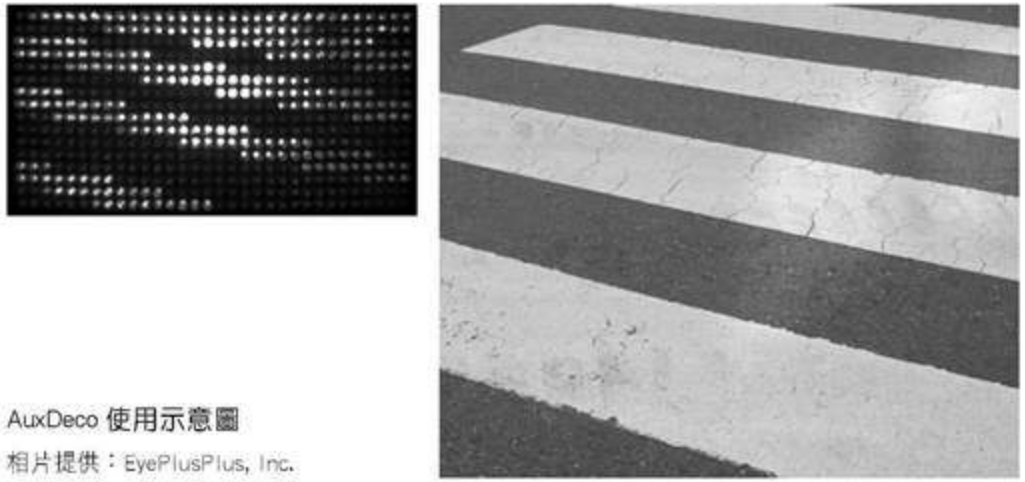
- BBC : Daniel kish , blind man use bat-like vision

第六个视频-
真实的蝙蝠
侠

从味觉产生视觉

首先在眼镜的镜片里安装微型数码相机，利用相机拍摄外部物体散射的光的信息，然后用“**Wicab BrainPort**”收集照相机拍摄到的信息并传输电极上，电极安放在到舌头两侧，呈“棒棒糖”状排列。 这种装置是专门为帮助盲人和视力低下的人设计的，可以让盲人和视力低下的人用“舌头”看见眼前的景象。

从触觉产生视觉



AuxDeco 使用示意圖

相片提供：EyePlusPlus, Inc.

是大脑看到了世界

- 不是眼睛看到了世界，是大脑看到了世界。
- 视网膜只产生了像素。
- 是大脑进行了极其复杂的处理，不但看到了物体轮廓，表面材质，还识别出了物体，人脸，并重建了一个虚拟3D现实世界。
- 请问大家这个视频是3D的吗？
- 如果不是，请你闭上一只眼睛，用一只眼睛看10秒钟。

视觉的关键特征

- 物体
  - 颜色，形状，纹理。
- 运动
  - 青蛙只对运动物体敏感。
- 场景
  - 深度关系，模式和预测

VR的眩晕问题

我们的视觉系统看外部世界的方式。当我们看近处的物体时，有6个主要的深度线索帮助我们形成三维视觉。

1、透视：远处的物体看起来更小；

2、遮挡：近处的物体会挡住处的物体；

3、双眼（立体）视差：左右眼看到同一物体的不同视图；

4、单眼（运动）视差：当头部运动时，远处和近处的物体会以不同幅度运动；

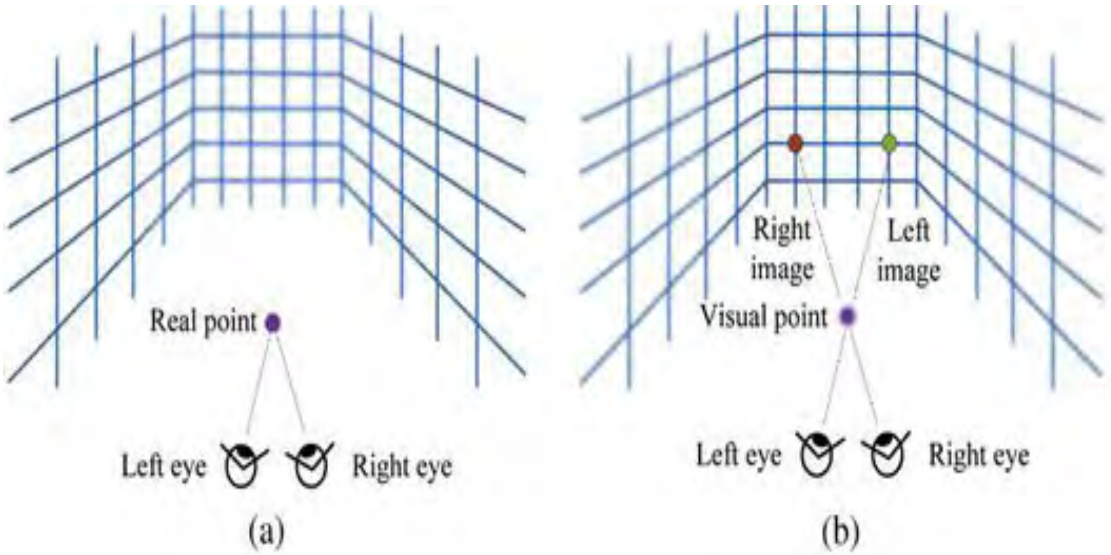
5、聚合：当眼睛聚焦在近处物体时两眼视线汇合；

6、调节：根据物体的距离，眼球相应地调整焦点。

最主要的三个因素

- 焦距测量，包括双目聚焦和单目焦点。
- 透视换算。
  - 双目聚焦到零视差物体时候不存在辐辏冲突。
  - 更远或者更近的物体虽然不符合辐辏关系，但是可以快速透视换算。
- 双目视差和运动视差。
  - 无论是哪一种视差，在平行聚焦的时候计算量都非常巨大。

焦点调节问题



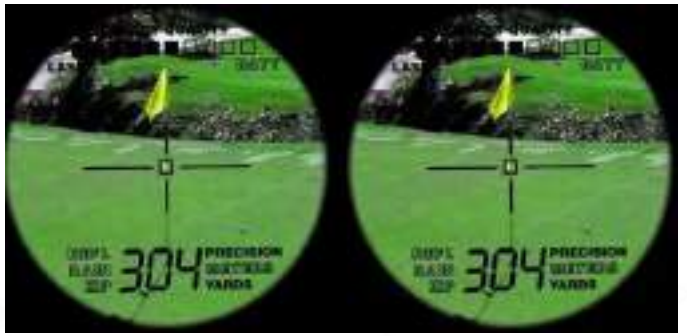
当你看一个物体时候，你会调整眼球。如果物体较近，眼球向内翻转，如果物体较远，眼球会向外，这样产生了视觉辐辏。与此同时，还会进行“适应性调节”也称焦点调节。通常，视觉辐辏与适应性调节是成对出现的，这是我们的生理现象。

戴上Oculus Rift或者**Gear VR**之后，之前的和谐状态全都被打破了。一般来说，**VR**设备会利用双眼视差（平行视差）来实现立体感。但这种技术会引起辐辏与适应性调节不一致，即我们说的视觉辐辏调节冲突。

第八个视频-LG-3D-DEMO

关键还是计算效率

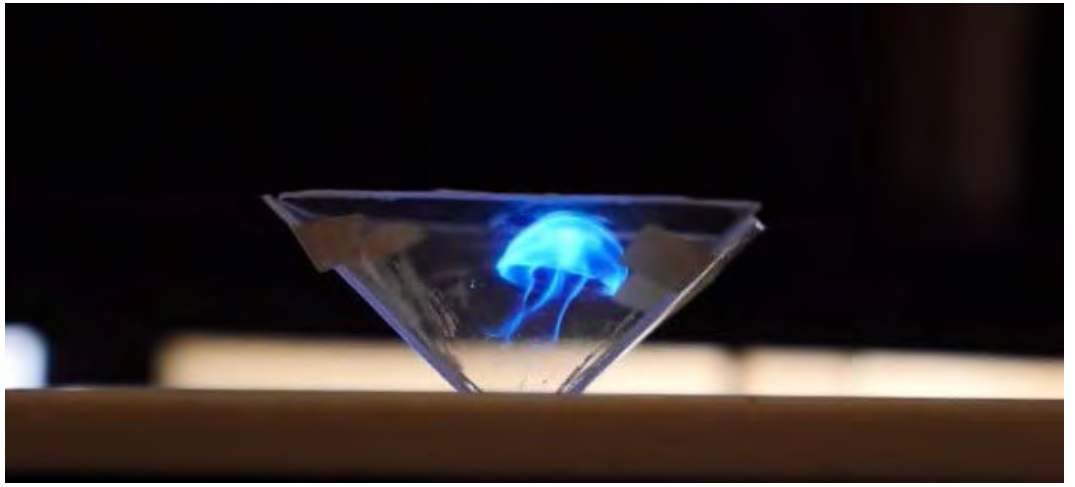
- 虽然平行视差的计算效率比较低，大脑还是可以适应的。
- 如果视野周围有一个稳定的参照光场，平行视差很容易被大脑换算过来。
- 独立的平行视差视野如果范围比较小的话，大脑也可以耐受。
- 全封闭沉浸式，超大视角和超大视野的平行视差是一个灾难。



降低眩晕感的办法

- 较小的视角
- 有限的视野
- 开放的VR设备
- 反射式MR

神奇的MR



演示视频

- **谢谢大家**

Thank You

XXXXX : 136000000

aaaa@bbbbbbbb.com

LiveVideoStackCon

聚音视　研修不止于形

关注LiveVideoStack公众号

回复 **鲍金龙** 为讲师评分