ClusTile:360度视频流的带宽最小化

# 潮州

SUNY·宾汉姆顿[czhou5@binghamton.edu](mailto:czhou5@binghamton.edu)

# 萧

乔治·梅森大学[mxiao3@gmu.edu](mailto:mxiao3@gmu.edu)

# 刘玥

SUNY·宾汉姆顿[yaoliu@binghamton.edu](mailto:yaoliu@binghamton.edu)

*摘要—360度视频有潜力通过为用户提供比标准流视频更沉浸式的交互环境来改变视频流体验。然而，这种体验受到高带宽需求的阻碍，高带宽需求是由与360度帧相关的额外信息产生的。因为用户看不到这个完整的360度视图，但是在大多数360度流系统中传输完整的视图，所以在这个领域有很大的潜力来减少浪费的带宽。*

我们提出ClusTile，这是一种分块方法，用于选择一组分块，当在预期的一组视图上流式传输360度视频时，允许使用最小的带宽。通过在收集的用户视图集群上独立求解一组整数线性规划(ILPs)来选择这些图块。聚类方法将迭代学习算法的计算要求降低到实际水平。与标准的360度流相比，从ClusTile计算出的镶嵌可以节省高达76%的带宽，与性能最佳的固定镶嵌方案相比，可以节省高达52%的带宽。

1. 介绍

360度视频为观众提供身临其境的视角，允许探索从摄像机位置发出的所有视角方向。日益复杂和不精确的硬件(例如三星的GearVR)支持360度视频体验。虽然可以在标准视频播放器上观看360度视频，但带有运动跟踪功能的头戴式显示器可以让这些360度视频真正身临其境。新硬件和360度视频的结合意味着它有可能在未来几年被广泛采用。尽管有这种潜力，但与标准视频流相比，360度视频流所需的软件和算法仍然相对不成熟。这些软件缺陷可能会阻碍主流的接受，因为视频流的带宽要求对许多用户来说可能太高了。

为了说明这个问题，虚拟现实设备上的360度视频流可能需要4K分辨率的等矩形帧，这是一种常见的360度编码，才能获得优质的观看体验[10]。由于综合因素，这种高分辨率是必要的。设备离用户眼睛很近，意味着观看时需要高分辨率。立体观看使传送的360度帧的分辨率减半，最后，每个360度帧只有一小部分显示在屏幕上。例如，以等矩形框架为中心的100×100度视图仅覆盖框架像素的14.3%。

网飞推荐的最低4K视频连接速度为25 Mbps [6]，但美国的平均宽带速度仅为15.3 Mbps [1]。这种带宽差异可能会给广泛的360度视频应用带来问题，而不会改善当前的现状。

360度视频流中至少有三个核心技术挑战与这个带宽问题有关:I)360度视图不能像标准视频那样直接编码在矩形帧中。ii)用户只能看到完整360度帧内的小视场，但这些完整帧通常是在流传输过程中传送的，浪费了带宽的未观看部分。iii)在流式传输期间，客户端不知道用户的未来方向，使得难以选择360度视图的区域。注意(iii)依赖于(ii)，因为只有当360度视频流基础设施能够支持反映预测的子区域的下载时，预测的用户取向才是有用的。

在这些问题中，这项工作的重点是第(二)项。具体来说，我们提出了一种平铺方案，我们称之为ClusTile，以最小化传送360度视频片段的观看部分所需的带宽量。通过“平铺方案”，我们指的是对投影的360度视频片段进行空间切片的方法，以允许片段的部分而不是整个360度视图的单个片段通过网络传输。请注意，在这些切片方案中，每个切片表示覆盖360度视图的一部分和(通常是固定的)时间间隔的动态自适应流(DASH)片段，而不是单个360度帧的一部分。（这篇文章就是提出tile切片方案）

平铺方法必须解决两个权衡才能有效运行。首先，随着图块大小的减小，视频编码效率会降低。一方面，切片方法必须选择足够小的切片，以覆盖360度视图的部分，而不会浪费太多像素。另一方面，图块大小不能太小，以至于浪费更少像素的好处被降低的编码效率所抵消。作为一个极端的例子，假设切片方案选择单像素切片。这些切片可以完全覆盖任何用户的视图，但是如果它们被独立编码，就好像没有应用压缩一样。切片大小的影响

第二个折衷涉及在使用更大数量的切片之间进行选择，从而消耗更大的存储空间，但是允许给定视图的更有效的流传输，

相对于为一个片段选择更少的切片，使用更少的存储空间，但减少了流传输期间用户视图的可能切片选项。为了说明这种折衷，假设我们在360度球体中以10度的间隔为所有10度桶存储渲染的100×100度视图。该方案将使用比单个等矩形图像更多的存储空间，但是将允许任何单独的视图被有效地传输。切片数量的影响

我们的ClusTile方法通过求解一个整数线性规划(ILP)来选择要在服务器端编码和存储的瓦片集，该规划被公式化以捕捉这些折衷的近似值。我们从一个更大的ILP中推导出ClusTile ILP，它更直接地代表了编码效率和存储的权衡，但无法实际解决。这个更大的ILP试图有效地覆盖从段视图数据集中收集的视图。ClusTile还受益于为进一步减少计算开销而设计的聚类过程。与在许多单个视图上运行单独的优化问题不同，对这些视图进行聚类通过在视图聚类上解决较少数量的优化问题来减少所需的计算。

ClusTile的一个潜在缺点是它生成的切片集可能会重叠，并且在流式传输时，没有简单的方法从这个集中选择切片来最小化网络带宽。然而，我们展示了对我们的整数程序的简单修改产生了一种快速的切片选择方法，该方法与缓存相结合，使得切片选择在流式传输期间变得实用。

评估显示，在对用户头部方向的完美预测下，与基线非平铺方案相比，ClusTile可节省高达76%的下载量，与性能最佳的固定平铺方案相比，可节省高达52 %的下载量。

1. 背景和相关工作

与传统视频流从单一视角编码视图不同，360度视频编码的是全方位视图。为此，首先将360度视频帧投影到2D平面上，然后使用标准视频编解码器(如H.264 [26])进行编码。

1. *360度视频投影和视图渲染*

虽然存在许多投影方案，但等矩形投影[2]是将360度框架投影到2D平面上最常用的方法。这种投影目前被YouTube和JauntVR等360度视频流媒体服务提供商使用。

在等矩形投影中，球体上的像素基于其偏航和俯仰角值被映射到2D矩形框(球体上的点的偏航和俯仰值是通过首先应用偏航运动然后应用俯仰运动来确定的)。在等矩形图像中，偏航角从左到右为-180度到180度，而俯仰角从90度到-90度

从上到下的度数。例如，球体上一个<偏航= 0，俯仰= 0 >的像素被投影到等矩形投影的中心。

在360度视频流期间，基于用户的头部方向(例如，在欧拉角<偏航、俯仰、滚转>)和视野(FOV)(例如，100度垂直和水平)在运行时呈现向用户显示的视图。对于目标图像中的每个像素，我们从球体中心向目标像素投射一条光线。这条射线将与球体上的一个点相交。然后，我们可以从上述光线与球体表面相交的点处的像素填充目标渲染图像中的像素值。（看来我的思路没有问题）

1. *360度视频流*

如今，360度视频流利用了HTTP动态自适应流(DASH)提供的比特率适配[18]。在DASH中，视频在时间上被分成多个片段。视频片段是在流式传输过程中作为单个单元下载的视频帧序列。片段长度可能有所不同，但它们通常由大约一秒或几秒钟的视频组成。每个段的多个版本以不同的比特率编码。随着时间的推移，随着可用带宽的波动，视频流客户端将选择比特率与可用网络带宽匹配的片段。该选择过程使算法能够在降低视频质量和停止(停止回放以缓冲额外的视频)之间灵活地进行权衡，从而允许在不同的网络条件下平滑回放。

然而，比特率适配不能解决360度视频流的根本低效问题。也就是说，并非视频帧中所有下载的像素都被用户消耗。虽然360度视频可以提供完整的360度水平视野，但目前大多数虚拟现实耳机的水平FOV只有90度到110度之间[9]。结果，360度视频帧中的许多像素没有被显示。例如，要呈现100度以<偏航= 0，俯仰= 0 >为中心的水平和垂直FOV度，在等矩形帧中仅需要大约14.3%的像素。网络带宽用于下载其余85%的像素因此被浪费。

为了解决这个问题，一种方法是设计一种定向投影方案，将360度视频帧中的更多像素用于球体上与用户视图方向匹配的区域。这种方案的一个例子是脸书提出的“偏移立方体投影”。然而，我们的测量结果表明，由于施加在球面上的失真，该方案的编码效率显著降低[28]。结果，即使与等矩形投影相比，该投影可以在不到一半的像素中产生相似的视觉质量，编码视频的比特率的降低也仅为5.6%至16.4%。偏置立方体投影的另一个缺点是，必须为许多方向编码单独的偏置立方体段，以支持所有方向的高质量。例如，脸书在其实现中使用了22个偏移立方体方向[28]。定向投影的许多版本显著增加了用于流式传输支持的视频的存储空间。

另一种方法是基于瓦片的流[14]，[15]，[16]，

[17], [19], [22], [23].图块是投影的360度视频帧上的矩形区域。切片是独立编码的，因此可以独立解码。基于切片的流通常与DASH一起使用，将DASH片段切割成多个切片，共同覆盖片段的整个帧区域。在一个理想的场景中，视频流客户端只需要下载覆盖用户视图的最小切片集。

平铺的一个挑战性问题是确定最能支持流应用程序的平铺大小。小的图块大小可以最大限度地减少用户无法观察到的通过网络传输的像素数量。然而，随着图块中像素数量的减少，视频压缩率也会受到影响[12]，[21]，[25]。出现这些降低的压缩率是因为运动估计(现代视频编解码器中帧间压缩的关键)在块匹配的搜索范围受限于小块时效率较低。因此，最小化切片大小的切片策略可能不会最小化下载带宽。另一方面，虽然大的图块大小可以提高单个图块的压缩率，但这些大的图块可能会导致更大数量的未查看像素。

现有的基于切片的流原型主要使用固定切片：视频帧被切割成不重叠的切片，每个切片的大小为切片宽度×切片高度[4]，[7]，[15]。在之前的工作中，我们提出了一种叫做OpTile的方法。OpTile优化了

兼顾服务器端存储成本和带宽成本的目标。这个优化问题的平铺解决方案是不重叠的，并且允许可变尺寸的平铺[27]。OpTile的一个缺点是它对非重叠切片的限制，这限制了带宽节省。

1. CLUSTILE的设计

ClusTile的目标是基于这样一种直觉，即360度视频流最重要的方面包括最小化用于向客户端传送视频流的带宽。我们假设每个片段都有一些潜在的视图分布，并且可以通过观察该片段过去视图的模式来有效地估计这种分布。因此，我们的目标依赖于过去视图的数据集。作者假设，对于每个全景视频片段，用户的视点分布是有规律的，并且可以通过片段过去的视点分布得到这个规律。

这种目标的解决方案显然包括选择对应于每个过去视图的图块，使得该图块尽可能精确地覆盖视图。同样明显的是，这种方法的问题是，随着过去视图数量的增加，切片的数量也将增加，以至于所有可能的几乎完全覆盖用户视图的切片都包含在切片解决方案中。但是，这种解决方案可能会导致实际问题，因为这些切片所需的存储量可能会超过可用的系统限制。

因此，我们试图找到一种切片解决方案，通过一组也可以存储在预定义空间量中的切片来最小化所需带宽。不幸的是，添加这个约束会产生一个即使对于相对较小的数据集也难以解决的ILP，因为ILP必须联合解决

表一:问题表述中使用的变量。

|  |  |
| --- | --- |
| *x* | 被选择并且编码用于流式传输图块集的矢量。 |
| *c* | 每块瓷砖下载成本的向量(即，带宽) |
| *M* | 矩阵将切片(列)映射到它覆盖的基本切片（行） |
| *x(v)* | 表示用于覆盖视图v的一组图块向量 |
| *b㈤* | 部分或全部被视图v覆盖的基本瓷砖。 |
| *x(k)* | 表示需要覆盖的一组图块的向量  视图簇k中的所有视图。 |
| *b(k)* | *b(k)是一个二进制向量，表示哪些基本瓦片包含在簇k中* |
| *n(k)* | 集群中视图数量的计数向量  *由向量的每个元素索引的瓦片覆盖的k。给定向量m(k)，其中m(k)表示*  基本图块被覆盖的次数，n(k) =  *j*  maxi(m(k) × Mij)。  *我* |

|  |  |  |
| --- | --- | --- |
| 基本tile0 | 基本瓷砖1 | 基本瓷砖2 |

图1:将一个片段segment分成三个基本块tiles。

对于所有数据集项。一个只有10个项目的数据集需要在商用机器上进行7分钟的计算，并且解决方案在数据集项目的数量上多项式地缩放。

为了克服这种计算限制，我们建议在数据集中的视图集群上运行我们的ILP，而不是在单个视图上运行。速度的提高是因为观察到，与ILP解决方案相比，使用标准工具可以非常快速地计算视图集群。此外，如果我们数据集中的许多视图非常相似，那么多次计算切片解决方案可能没有用。将这些相似的视图组合在一起是解决这种计算低效的自然方法，因为我们期望相似的视图产生相似的切片覆盖解决方案。

*A.问题定式化*

要解决找到最佳切片解决方案的问题，我们必须首先创建一组候选切片。为了创建这个候选集，我们使用了我们在OpTile [27]中提出的相同方法。我们在这里描述这个方法，这样问题的表述就完整了。

给定一个视频片段，我们首先构建一组基本切片tile。每个基本tile都是在视频流期间可以编码和下载的最小图块tile。任何有效的图块tile都必须是矩形，并且完全覆盖一个或多个基本图块tile。例如，如果我们如图1所示将一个视频片段在空间上划分为3个基本切片，那么总共有6个可能的tile可以由这些基本tile创建，如图2所示。（终于懂了在干嘛，就是将每个segment预分割成小块basic-tile，然后组合这些基本块为最终的tiles）我们可以用一个二进制向量x来编码解中这些可能瓦片的存在:[a，b，c，d，e，f ]。例如，如果决定将线段垂直切割成两个不相交的图块a和e，那么x将表示为[1，0，0，0，1，0]。

瓷砖a

基本平铺0

瓷砖b

基本瓷砖1

|  |  |  |
| --- | --- | --- |
| 基本平铺0 | 基本瓷砖1 | 基本瓷砖2 |

瓷砖f

瓷砖c

基本瓷砖2

瓷砖d

基本瓷砖2

基本瓷砖1

基本瓷砖1

基本平铺0

瓷砖e

知道个图块的成本，其中n是基本图块的数量。为此，可以对切片进行编码来计算这些存储空间

成本，在重大的计算费用，或这些成本可以估计。估计要求估计误差足够小，或者将真正的块大小限制得足够紧密，这样ILP解决方案仍将具有可接受的性能。

第二个也是更难克服的问题是，这个整数线性程序的大小与数据集中的项目数成正比。这些整数线性

图2:如果片段被分成3个基本瓦片，则可以构建六个可能瓦片的候选集合。

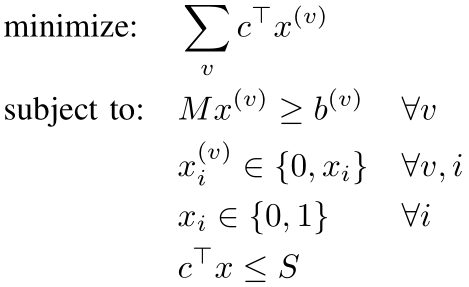
与OpTile [27]只考虑非重叠瓷砖不同，在ClusTile中，可以选择重叠瓷砖。例如，如果x = [0，0，0，1，1，0]，选择切片

*d和e，则基本瓦片1被覆盖两次。*

我们使用向量c来表示下载每个图块所需的带宽量。该带宽量可以通过对相应的区块进行编码来估计或直接计算。

我们制定了以下整数线性规划(ILP)目标，旨在找到一个用于流式传输360度视频片段的最佳切片集。该目标旨在最小化用于流式传输该段视图的预期网络带宽，假设在这段视频上视点固定分布，受存储限制。这里，最佳的瓦片集合被编码在x的值中。

*联合带宽最小化目标:*

即使对于小数据集，程序也变得不可行。例如，十个视图的数据集在商品硬件上求解大约需要400秒。

我们提出了一个联合带宽最小化目标的易处理近似。这个近似值的解决方案可以在商品硬件上的实际时间内计算出来，但可能无法实现原始目标的精确保证。

*ClusTile目标:*

最小化:c⊤diag(n(k))x(k)受制于:Mx(k) ≥ b(k) ∀k

*x(k)∞{ 0，1} ∀k，I*

σI

*x(k) ≤ N ∀k*

*我*

*我*

ClusTile目标类似于联合带宽最小化目标，在一组视图上定义了一个综合目标。主要有四个区别。

首先，它不是试图涵盖单个视图，而是试图涵盖视图集群。这里，向量b(k)是

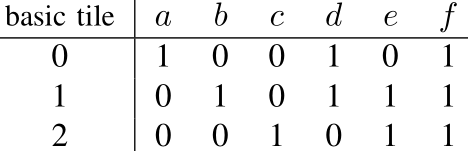
一个二进制向量，指示哪些基本瓦片包含在簇k中:

*b(k) =*

*v∈k*

*b㈤*

在联合带宽最小化目标中，对于片段的视图v，b(v)是二进制向量，其中1表示视图至少部分显示了对应位置的基本图块。x(v)是x中的图块子集，用于覆盖b(v)中的所有基本图块，如视图v所要求的。此覆盖约束使用矩阵M表示*。*M是一个二进制矩阵，其行代表基本瓦片，其列代表瓦片。M的位置(i，j)处的值1表示由列j表示的图块覆盖了由行i表示的基本图块。例如，图2中示例的M的内容如下所示:



联合带宽最小化目标在整个视图数据集上最小化流传输中使用的带宽，但受x中所有切片的总存储大小的约束。然而，这一目标有两个实际问题。首先，为了计算成本向量c，我们需要

如果簇k中的任何视图包括一个基本图块，那么我们在b(k)中通过相应向量元素中的1来指示这个基本图块。但是，我们注意到集群上的其他阈值

内容对于构造b(k)是可能的。具体来说，由少量簇元素覆盖的基本瓦片可以从b(k)中排除。

第二，与联合带宽最小化目标不同，ClusTile目标中的成本向量由给定图块覆盖的集群中的视图数量进行缩放，该权重由向量n(k)给出。

第三，我们在每个集群的解决方案xk中包含一个约束，要求xk中不超过N个非零项。也就是说，不应该使用超过N个图块来渲染集群k所需的视图。此约束基于以下实际问题:对于H.264，每个图块都必须独立解码。这意味着对于渲染视图所需的每个图块，必须启动一个解码器实例。所有解码器都必须及时完成解码任务。如果实例太多，这可能会导致高上下文切换开销。解码的流也必须同步

在视图呈现发生之前。如果渲染视图需要太多的切片，这种同步开销会很大。

最后，ClusTile目标删除了交叉视图约束x和总存储大小约束c⊤x ≤ S。删除这些交叉视图约束允许我们解决

*独立地为每个集群设定目标，提高ILP的易处理性。综合解决方案，x，适用于所有人*

|  |  |
| --- | --- |
| σs  *i∈t i* | 瓷砖内所有基本瓷砖的总储存尺寸t如果  它们都是独立编码的。 |
| σmv  *i∈t 我* | 需要的运动矢量总数  如果瓦片t中的所有基本瓦片被独立编码，则重新定位。 |
| *mvt* | 如果块t  是独立编码的。请注意，该集合确实  不包括描述在图块t中找到的基本图块之间的运动矢量的mvi。 |
| *o* | 每个重新定位的mo的平均存储开销  该段中的矢量。 |
| *新约（new testament的缩写）* | t内基本瓷砖的数量。 |

表二:用于瓷砖尺寸估算的特征。

集群被构造为x = k xk。尽管缺少

总切片大小限制，我们希望所有集群都有解决方案

只要集群数量有限，就要消耗合理的空间。

ClusTile Objective还通过将多个视图包含在单个集群视图中来减少整数线性程序的大小，从而提高了计算的可处理性——在每个视图的基础上，集群的计算速度比ILP解决方案快得多。这种方法的一个潜在缺点是，如果集群具有高方差，那么为集群计算的覆盖范围将导致在流式传输单个视图时浪费带宽。也就是说，如果聚类不能紧密地表示视图组，那么从由聚类目标提供的集合中选择瓦片来覆盖单个视图的最佳方式也可能经常覆盖360度球体的未被观察的区域，使得该聚类目标效率低下。我们试图在第五节中根据经验评估这种聚类近似的效果

最后一个问题是，我们注意到这些目标的解决方案可能会导致这样的情况，即我们无法在解决方案中使用图块覆盖训练集之外的视图。也就是说，我们的目标的解决方案可能无法在一组抽样视图之外很好地推广。一个可能的“正则化”策略可能包括总是在解决方案中包含一组基本的瓦片。这种方法将确保所有在训练集中看不到的视图都是可覆盖的，但是要付出额外存储的代价。我们在第节的实验中使用了这种方法

动词 （verb的缩写）

当前块和最佳匹配块之间的差值，并压缩该差值(残差)。如果当前块和参考块高度相似，则只需要少量数据来编码残差。将帧切割成小块减少了视频编码器可用的潜在块匹配的数量。结果，对于更小的块，可以定位不太相似的参考块并将其用于运动估计，导致更大幅度的残差不能被有效地压缩。

我们的成本估算是在视频片段上完成的。每个片段包含一小段视频，通常为一到三秒，并且只能包含一组图片(GOP)。也就是说，每个片段可以包含一个单独的帧内编码的I帧。我们将未平铺的原始片段的存储大小表示为s。然后，我们在空间上将该片段分割成

选定尺寸的基本瓷砖。基本图块的大小必须是H.264宏块的倍数，即16 × 16。我们的预测方法从编码所有的基本瓦片开始。我们将第I个基本切片存储大小表示为si。然后，该方法提取所有运动矢量

并计算如果我被独立编码的话，有多少指向一个基本块的运动矢量需要被重新定位。我们把这个计数称为mvi。然后，预测过程计算每个重新定位的存储开销

。

*B.估价*

运动矢量为o =

σI si s

我mvi

我们的整数规划公式的一个重要部分是与x中的每个项目相关联的成本c。这里，ci表示与tile xi相关联的段的存储大小。考虑到

*n个基本瓦片中，存在O(n2)溶液瓦片。如此大量的瓦片使得对每个瓦片进行编码以获得其大小成为可能*

计算量太大，不实用。

相反，我们通过一种预测策略来估计每个可能图块的存储大小，该策略将视频压缩和运动估计之间的强相关性考虑在内。为此，我们使用了之前在[27]中提出的方法。我们在这里描述这种方法，以便这一部分是独立的。

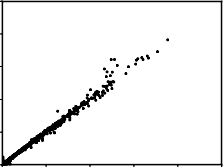
今天的视频编解码器通过基于块的运动估计利用帧间相似性进行压缩。为了对块进行编码，视频编码器在附近的帧中搜索最佳匹配的块。这个最佳匹配块与被编码块的相对位置随后被记录为“运动矢量”。然后，视频编码器计算

给定一个包含多个基本图块的图块t，我们的

预测方法使用表二中显示的五个特征来估计t的存储大小。我们使用6082个切片的数据集训练了一个人工神经网络模型。为了生成这些切片，我们首先在时间上将视频切割成一秒钟长的片段。然后，我们从视频中随机选择一个片段，一个平铺位置，以及平铺的宽度和高度。一旦选择了一个图块，我们就从表二中提取特征，并使用ffmpeg [3]对图块进行编码，以获得以字节为单位的真实图块大小。我们将我们的人工神经网络配置为使用具有ReLU激活功能的50个节点的单个隐藏层。L-BFGS [20]被用于训练人工神经网络。

图3显示了我们四重交叉验证的预测结果。在每个折叠中，我们基于来自三个视频的特征和经验瓦片大小来训练人工神经网络模型。然后，我们在第四个视频的切片上评估了训练好的模型。预测的总中值绝对误差为4.7%，在四次折叠中从2%到8%不等。

我们使用了神经网络模型，用

500

真实切片大小(千字节)

400

300

200

100

0

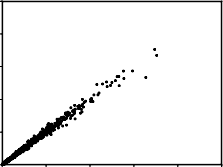
0 100 200 300 400 500

预测切片大小(千字节)

1. 视频1

500

400



真实切片大小(千字节)

300

200

100

0

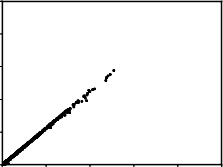
0 100 200 300 400 500

预测切片大小(千字节)

1. 视频2

500

400



真实切片大小(千字节)

300

200

100

0

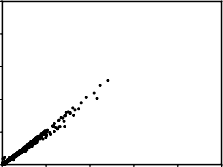
0 100 200 300 400 500

预测切片大小(千字节)

1. 视频3

500

400



真实切片大小(千字节)

300

200

100

0

0 100 200 300 400 500

预测切片大小(千字节)

1. 视频4

图3:四重交叉验证的结果。在每个折叠中，我们基于来自三个视频的特征和经验瓦片大小来训练神经网络模型。然后，我们在第四个视频的切片上评估了训练好的模型。

用于我们在第五节中评估的图块大小(即矢量c)估计的数据集。用于ANN训练的这一组四个视频与第五节中用于带宽缩减评估的五个视频不重叠

*C.流式客户端切片选择*

由于每个基本图块可能被x中的图块覆盖不止一次，因此需要图块子集选择算法来选择给定x和b(v)的x(v)。

*客户端切片选择目标:*

⊤

片段被切割成固定大小的图块，每个图块编码n × n个像素。我们还将ClusTile的带宽节省与我们之前的工作OpTile [27]进行了比较。

我们的ClusTile目标需要每个细分市场的一组现有视图，以及这些现有视图的基本图块覆盖范围b(v)。为此，我们使用了一个公开可用的数据集:360度视频头部运动数据集[13]。这个数据集记录了58个观看的用户的头部方向

5个单视场360度视频，编码在等角投影中，在虚拟现实耳机中。我们下载了这5个视频，并以两种分辨率重新编码:1920×960和

最小化:c(r)

*x(v)*

3840×1920.对于数据集中的每个视频，用户都会观看每个视频

服从:Mx(v) ≥ b(v)

*x(v)∞{ 0，xi} ∀v，I*

*我*

对于x中的所有I，其中xi为1，c(r)代表真实存储

视频约80秒。我们提取了这80秒

并把它们编码成一秒钟的片段。

在这个片段提取步骤之后，我们获取每个片段，并构建一组基本切片。对于1920 × 960-

瓷砖的尺寸

*我*

是在中的所有图块之后根据经验获得的

分辨率视频，我们将基本平铺设置为包含4 × 4 H.264

*x已经被编码了。这个整数程序然后求解给定x，M，b(v)和c(r)的最优x(v)。由于x中非零项的数量通常很少，我们可以将x(v)的元素和M的列减少到只对应于x的非零元素，从而允许整数线性程序非常快地求解。我们在第五节中报告了解决时间*

1. 履行

对于每个视频片段，我们使用一组已知的d(v)构造ClusTile目标，并使用ANN模型估计成本向量c。当构造矩阵M时，我们限制每个图块的高度和宽度，这样一个图块最多可以包含

12 × 12基本瓷砖。我们使用现成的K-means [8]实现来聚集视图，并使用Gurobi Optimizer

7.0.2 [5]求解整数线性规划。

*然后使用ffmpeg基于ClusTile解决方案对切片进行裁剪和编码。为了保持所有切片的相同质量水平，我们使用相同的x264参数对所有裁剪的切片进行编码。对视频中的所有片段重复相同的过程。*

1. 估价

我们将ClusTile与两种基线360度流解决方案进行了比较:i) origin，一种不使用切片的解决方案，以及ii) fix-n，一种固定切片方案，其中

宏块，即64 × 64像素。对于3840 × 1920的视频，

我们使用更大的基本图块，128 × 128像素，以减少x中的解决方案图块数量。这些尺寸选择意味着

对于两种视频分辨率，基本图块的数量为

30 × 15 = 450.

一旦产生了基本的切片，我们从数据集中随机选择40个用户作为训练集，并为每个时间片段提取他们的视图方向。用户的观看方向可以随着一个片段的回放而改变，例如，用户可以在1秒的片段持续时间内扫视视频。为此，头部运动数据集在一秒钟内记录多个瞬时视图方向样本。为了构造b(v)，我们假设一个100度水平和100度垂直视场的视口。对于用户u在一个片段上的视图v，如果在渲染同一片段中用户u的任何瞬时视图时使用了基本图块I中的任何像素，我们将b(v)设置为1。我们重复这个策略来为每个片段构建40个b(v)的集合。然后我们将这些视图聚类成K个聚类。我们实验了两种不同的K设置:5和10，为ClusTile物镜创建了5或10个b(k)。我们将这些设置称为ClusTile-5和ClusTile-10。对于每个簇k，我们将覆盖簇中所有基本瓦片b(k)所需的瓦片的最大数量限制为10个。

*我*

为了根据我们的目标构建向量c，对于每个片段，我们估计每个可能的区块的存储大小

表三:ClusTile解决方案中瓷砖的平均数量。

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | 1920 × 960 | | 3840 × 1920 | |
| 意思是 | 中位数 | 意思是 | 中位数 |
| ClusTile-5 | 46.3 | 50 | 47.7 | 50 |
| ClusTile-10 | 85.4 | 89 | 91.4 | 95 |



图4:ClusTile-5解决方案生成的图块。一个区域越暗，对应的基本图块被覆盖的次数就越多。

使用第三章b节中描述的人工神经网络模型

求解ILP后，我们根据x中编码的切片将每个片段切割成切片。我们使用ffmpeg对x中的每个切片进行编码。我们从这些真实的编码存储大小构建向量c(r)来评估客户端切片选择时间。

1. *服务器端切片存储大小*

由于我们将渲染一组视图所需的最大切片数设置为10，因此

ClusTile解决方案是K ×10，其中K是我们在ClusTile目标中使用的集群数量。也就是ClusTile-5

解决方案最多可以包含50个切片，ClusTile-10解决方案最多可以包含100个切片。表三显示了ClusTile解决方案中两种分辨率下视频的平均和中间切片数。图4显示了ClusTile-5解决方案中所选图块的两个示例。我们使用灰度来表示片段中的基本图块被选定图块覆盖的次数。一个区域越暗，其对应的基本图块被解决方案中的图块覆盖的次数就越多。

对于每个片段，我们对ClusTile解决方案中的所有切片进行求和，并针对原始的、非切片的等矩形片段对求和进行归一化。图5显示了结果。对于两种分辨率的片段，ClusTile-5的归一化存储大小的中位数仅为2.3。对于ClusTile-10，由于生成了更多的切片，其归一化存储大小的中位数略大，为3.6。如果我们应用“正则化”策略，该策略包括解决方案中的所有基本瓦片

对于1920 × 960，服务器端存储大小增加了2.1倍

视频，3840 × 1920的视频提高了1.3倍。

总的来说，ClusTile使用的存储空间是原来的2.3到5.7倍

尺寸。与脸书使用的偏移三次投影方案相比，这要小得多。在他们的方案中，使用了22个偏移立方体方向，要求每个段以22个版本编码，这大大增加了服务器的存储需求。与非平铺的等矩形数据段相比，ClusTile用增加的服务器端存储来换取更高效的下载。接下来，我们展示带宽减少的结果。

1. *流传输过程中的带宽减少*

流式带宽节省依赖于预测用户的未来视图，并且仅下载视图渲染所需的视频片段的部分。尽管视图预测不在本文讨论范围之内，但最近的研究表明，在不久的将来，用户的头部运动是高度可预测的[11]，[24]。在本研究中，我们评估了ClusTile在两种不同预测方案下的性能:完美预测和天真预测。由于我们使用了来自40个用户的视图方向来构建ClusTile目标，因此我们评估了其余18个用户的网络性能。

* 1. *完美预测:如果我们能够完美预测用户如何看待一个细分市场，我们首先评估ClusTile是如何形成的。完美预测允许我们下载最小的可用图块子集，每个图块都用于视图渲染。结果如图6所示。该图中的误差线代表95%的置信区间。ClusTile-10始终表现最佳，与下载完整、非平铺、等矩形片段的基线origin相比，下载量减少了68%至76%。ClusTile-5的性能比ClusTile-10稍差，节省了66%到73%的下载量。ClusTile-5和ClusTile-10的性能都优于所有fix-n方案。与性能最好的fix-n方案相比，ClusTile-10可以节省19%到52%的下载量。与OpTile1相比，ClusTile-10可以产生额外的7%至*

下载数据节省21%。

* 1. *朴素预测:我们接下来评估当朴素预测方案用于视图预测时的带宽效率。在这个天真的预测方案中，我们假设用户在第二秒的视图将匹配他或她的视图*

间隔[s + 3，s+4]。这里，在时间s的视图vs是在秒s开始时的瞬时视图，而

用户可以在1秒钟内扫视视频。我们进一步假设完美的预测可以在片段播放前一秒做出。假设每个片段有一秒长，这个完美的预测将是

能够准确预测在时间s + 2如何看待片段s + 3，即v[s+3，s+4]。因此，在这个天真的预测过程中的时间s + 2，我们允许附加的瓦片被

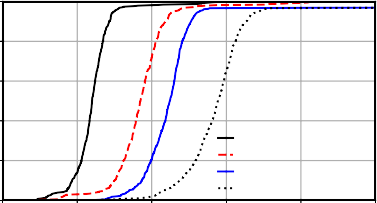
下载以涵盖在时间s未下载的基本图块，也就是说，在时间s+2，我们在bvmissing下载图块，其中bvmissing = bv[s+3，s+4) ⊕ (bv[s+3，s+4) ∧ bvs)。这样，一个片段的总下载数据就是下载的切片的大小

覆盖bvs加上下载的图块大小，以额外覆盖bvmissing。

结果如图7所示。在所有视频和两种分辨率下，ClusTile-10始终表现最佳，与origin相比节省了49%至63%。ClusTile-5表现第二好，与origin相比节省了46%到59%。与性能最佳的固定

1OpTile使用参数α来控制存储和带宽成本的相对重要性[27]。α越大，带宽节省越多。在[27]中，我们测试了α = 0、1和1000的OpTile，α = 1000产生最佳结果。因此，在本文中，我们将ClusTile与OpTile α = 1000进行比较。

1.0



ClusTile-5 ClusTile-10 ClusTile-5+基本型

ClusTile-10 +基本版

累积分数

0.8

0.6

0.4

0.2

0.0

0 2 4 6 8 10

(a) 1920 × 960

2.0

1.5

标准化下载量

1.0

0.5

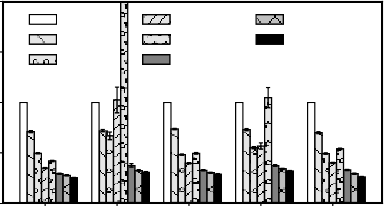
0.0

潜水巴黎r-coast延时威尼斯

(a) 1920 × 960

2.0

1.5



原点定位-512

fix-256

fix-128 fix-64

OpTile

ClusTile-5

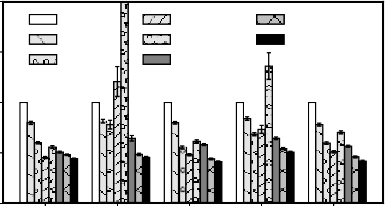
ClusTile-10

标准化下载量

1.0

0.5

0.0



原点定位-512

fix-256

fix-128 fix-64

OpTile

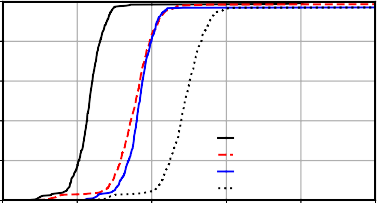
ClusTile-5

ClusTile-10

潜水巴黎r-coast延时威尼斯

(a) 1920 × 960

1.0



ClusTile-5 ClusTile-10 ClusTile-5+基本型

ClusTile-10 +基本版

累积分数

0.8

0.6

0.4

0.2

0.0

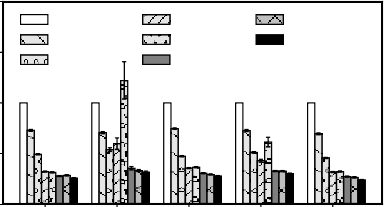
0 2 4 6 8 10

3840 × 1920

图5:标准化服务器端的CDF

2.0

1.5



起源

fix-1024 fix-512

fix-256

fix-128 OpTile

ClusTile-5

ClusTile-10

标准化下载量

1.0

0.5

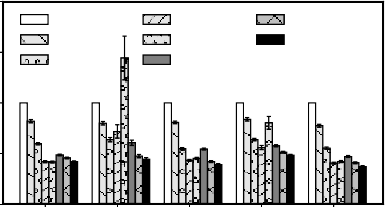
0.0

潜水巴黎r-coast延时威尼斯

3840 × 1920

2.0

1.5



起源

fix-1024 fix-512

fix-256

fix-128 OpTile

ClusTile-5

ClusTile-10

标准化下载量

1.0

0.5

0.0

潜水巴黎r-coast延时威尼斯

3840 × 1920

各种切割方案的存储大小。在此图中，“+基本”表示解决方案中还包括所有450个基本切片时的标准化存储。

图6:每段标准化向下

负载量平均来自18个用户的轨迹，具有完美的预测。

图7:每段标准化向下

负载量平均来自18个用户的原始预测。

表四:渲染一个片段所需的平均切片数。

|  |  |  |
| --- | --- | --- |
|  | 1920 × 960 | 3840 × 1920 |
| ClusTile-5 | 6.8 | 7.6 |
| ClusTile-10 | 7.0 | 8.1 |
| 固定-64 | 106.7 | 不适用的 |
| fix-128 | 32.2 | 106.7 |
| fix-256 | 9.7 | 32.2 |
| fix-512 | 2.2 | 9.7 |
| fix-1024 | 不适用的 | 2.2 |

平铺方案中，ClusTile-10的表现至少与3840 × 1920中的“潜水”一样好，最多可以节省39%(1920×960中的“巴黎”)的下载量。与OpTile相比，ClusTile-5可以节省5%到25%的额外下载

体积，而ClusTile-10可以额外节省13%到29%。

1. *客户端切片选择*

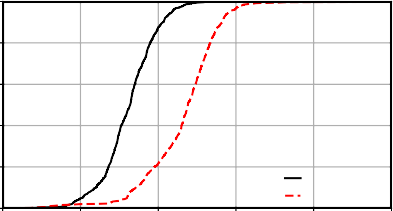
我们在表四中进一步显示了渲染一个片段所需的视图所需的平均切片数。使用ClusTile生成的图块，平均需要的图块不到9个。这比除了一个固定瓷砖之外的所有瓷砖都要小

方案:fix-512适用于1920 × 960视频，fix-1024适用于3840 × 1920视频。然而，fix-512和fix-1024切割需要更多下载带宽的大块。

最后，我们报告客户端图块选择目标的解决时间。在我们的实验中，我们使用了英特尔酷睿i5-6600 3.30千兆赫的中央处理器。只有一个单一的中央处理器核心用于解决ILP。结果如图8所示。ClusTile-5的平均求解时间为0.132秒。

1.0

0.8



usTile-5 usTile-10

氯氯

累积分数

0.6

0.4

0.2

0.0

0.10 0.12 0.14 0.16 0.18 0.20

客户端切片选择时间(秒)

图8:客户端图块选择时间分布。

对于ClusTile-10，由于解决方案包含更多的图块，平均求解时间稍长，为0.146秒。

1. 讨论和未来的工作

在本节中，我们首先讨论我们的算法在观看360度视频时对反映用户行为的数据的强依赖性。这种强烈的依赖性可能会让我们考虑，如果数据缺失或不符合某些预期，会出现什么问题。例如，在没有任何用户数据的情况下，ClusTile将如何运行？同样，如果一个细分市场上的视图分布随着时间的推移而变化，会怎么样？

我们预计，如果我们检测到视图在基本图块上的分布发生了变化，通过简单地重新计算图块集x，应该可以解决这两种情况。这些分布变化可以通过对基本切片视图计数的归一化向量之间的距离应用阈值来检测。该距离阈值越小，需要的计算就越多，因此可以根据可用资源来调整图块重新计算过程的灵敏度。

对于没有用户数据可用的情况，ClusTile可以选择固定的切片方案。在收集了更多经验用户视图并生成了更好的切片后，可以替换这些固定切片。

使用分块片段的360度视频流中的一些重大困难涉及预测未来视图的策略和给定这些预测的最佳选择要下载的分块的策略。如果下载策略选择的切片和用户实际查看的切片之间存在差异，那么消耗历史用户视图的ClusTile目标可能不是最优的。例如，如果用户的视图仅覆盖图2中的基本图块0，但客户端选择图块a和b进行下载，那么服务器端的图块选择算法(例如ClusTile)可能应该在解决方案中包含图块d，而不是图块a和b，因为下载图块d可能比下载图块a和b更有效。未来的工作包括更好地描述360度流期间的不确定性来源，并开发图块选择算法来解决这一问题。

1. 结论

在这项工作中，我们提出了ClusTile，一种计算360度视频的方法。ClusTile的灵感来源于最小化用于流式传输360度视频的带宽的需求，假设用户的浏览方式以可以通过记录的用户行为来表征。ClusTile还试图在其带宽最小化解决方案中限制存储切片所需的空间量。

我们首先提出了一个目标，即在总存储空间受到直接限制的情况下，共同尝试在过去的用户视图上最小化带宽。不幸的是，这个目标在大规模优化上是不切实际的。然后，我们通过对用户视图进行聚类并在这些聚类上求解独立的迭代学习算法来提高我们方法的计算效率。将这些解决方案结合在一起，就产生了一组切片，可以在流式传输过程中有效地呈现360度片段。

这种效率表现在，与非平铺等矩形段相比，我们节省了高达76%的带宽，与性能最佳的固定平铺方法相比，节省了高达52%的带宽。对于性能最佳的ClusTile配置ClusTile-10，其解决方案消耗的存储空间从第10百分位的非平铺等矩形部分的3.0倍，到第90百分位的4.3倍不等。

1. 确认

我们感谢匿名参考的建设性意见。这项工作得到了国家自然科学基金CNS-1618931的部分资助。

参考

1. Akamai的[互联网现状]2016年第一季度报告。https:

[//www.akamai.com/uk/en/multimedia/documents/state-of-the-internet/](http://www.akamai.com/uk/en/multimedia/documents/state-of-the-internet/)akamai-state-of-the-internet-report-q1-2016.pdf。

1. 等矩形投影。<http://mathworld.wolfram.com/>EquirectangularProjection.html。
2. FFmpeg。[http://www.ffmpeg.org/.](http://www.ffmpeg.org/)
3. Gpac hevc基于瓦片的自适应指南。https://gpac.wp.imt.fr/2017/02/ 01/hevc-基于瓷砖的适应指南/。
4. 古罗比。[http://www.gurobi.com/index.](http://www.gurobi.com/index)
5. 互联网连接速度建议。https://help.netflix.com/ en/node/306。
6. 克瓦扎尔。https://github.com/ultravideo/kvazaar.
7. sklearn.cluster.KMeans。<http://scikit-learn.org/stable/modules/generated/>sklearn.cluster.KMeans.html。
8. 虚拟现实耳机对比图。https://freeflyvr.com/虚拟现实-耳机-对比-图表/。
9. 虚拟现实(vr)和360视频101 -初学者指南。https://medium.com/visbit/虚拟现实-VR-and-360-videos-101-a-初学者指南-70bbade8e39。
10. Y.包，吴洪浩，张铁生，拉姆利，刘。拍摄运动目标:基于运动预测的360度视频传输。2016年IEEE大数据国际会议(大数据)，第1161-1170页，2016年12月。
11. C.Concolato，J. Le Feuvre，F. Denoual，E. Nassor，N. Ouedraogo，和

J.塔凯。基于mpeg-dash的hevc拼接视频自适应流式传输。

*IEEE视频技术电路和系统会议，2017。*

1. X.德·西蒙尼和西蒙。360度视频头部运动数据集。在ACM MMSys 2017，2017。
2. 长度达孔托、范登贝尔格、托马斯和尼亚穆特。使用mpeg dash srd可缩放和导航视频。《第七届国际多媒体系统会议录》，第34页。ACM，2016年。
3. 米（meter的缩写））格拉夫、蒂莫勒和穆勒。通过http实现带宽高效的全向视频自适应流:设计、实现和评估。在第八届多媒体系统会议记录，第261-271页。ACM，2017年。
4. 米（meter的缩写））侯赛尼。360°虚拟现实视频流中的视图感知平铺适配。在虚拟现实中，2017年IEEE，第423–424页。IEEE，2017。
5. 米（meter的缩写））Hosseini和V. Swaminathan。基于mpeg-dash srd的自适应360 vr视频流。在多媒体(ISM)中，2016年IEEE国际研讨会，第407–408页。IEEE，2016。
6. 国际标准化组织/IEC 23009-1:2014信息技术–通过HTTP (DASH)的动态自适应流式传输–第1部分:媒体呈现描述和片段格式。标准，国际标准化组织，2014年5月。
7. J.勒夫尔和康科拉托。使用mpeg-dash的基于切片的自适应流。《第七届国际多媒体系统会议录》，第41页。ACM，2016年。
8. D.刘和诺奇达尔。大规模优化的有限内存bfgs方法。数学编程，45(1):503–528，1989。
9. K.Q. M. Ngo、R. Guntur和W. T. Ooi。基于用户访问模式的可缩放视频流自适应编码。第二届多媒体系统年度会议论文集，第211-222页。ACM，2011年。
10. D.奥其，久尼塔，藤井，小岛康誉，岩木和广濑。Hmd观看球形视频流系统。第22届美国计算机学会国际多媒体会议录，第763-764页。ACM，2014年。
11. D.奥其、库尼塔、卡姆达、小岛康誉和岩木。全方位视频直播系统。在虚拟现实中，2015年IEEE，第349-350页。IEEE，2015。
12. F.钱，纪立群，韩斌，和戈帕拉克里希南。通过蜂窝网络优化360°视频传输。《第五届万物细胞研讨会论文集:操作、应用和挑战》，第1-6页。ACM，2016年。
13. 名词（noun的缩写）广明·希姆、g .拉温德拉、a .卡里尔和W. T. Ooi。支持动态感兴趣区域裁剪的可缩放视频流。第一届多媒体系统年度会议论文集，第259-270页。ACM，2010年。
14. T.威根德、沙利文、比约恩加德和鲁特拉。h. 264/avc视频编码标准概述。IEEE视频技术电路和系统学报，13(7):560–576，2003。
15. 米（meter的缩写））肖，周春华，刘玉英，陈素珊。Optile:在360度视频流中实现最佳平铺。在2017年多媒体会议会议记录中。ACM，2017年。
16. C.周，李，刘。Oculus 360度视频流的测量研究。第八届国际多媒体系统会议录。ACM，2017年。